Azure网站 Web Apps WebSites 云计算 Azure Web Apps



Azure WebSites 权威指南

微软云计算Web平台开发实战详解

赵伟 编著

- 微软专家级工程师多年经验总结
- 深入揭示Azure网站（Azure Web Apps）实用技术
- 涵盖Web应用架构、开发、部署、迁移和维护完整生命周期
- 基于Azure网站设计、开发和部署高性能、高可靠Web应用



清华大学出版社

Azure网站 Web Apps WebSites 云计算 Azure Web Apps

Azure WebSites 权威指南

微软云计算Web平台开发实战详解

赵伟 编著



北　京

内 容 简 介

本书介绍微软平台即服务模式的云计算平台产品 Azure 网站。全书共 8 章。第 1 章介绍云计算和微软云计算平台，并深入介绍 Azure 网站的基本架构设计和主要概念。第 2 章介绍 Azure 网站的管理、配置和监视，第 3 章介绍 Azure 网站管理自动化。第 4 章至第 8 章涵盖了从 Azure 网站架构、开发、部署到迁移的整个 Web 应用生命周期，以及基于 Azure 网站构建高性能 Web 应用的问题，并讨论了使用 Kudu 站点、诊断即服务、应用配置转换等高级专题。

本书可帮助 IT 管理员、开发人员和架构师深入了解 Azure 网站，也可供个人开发者学习如何开发和部署基于 Azure 网站的 Web 应用。

图书在版编目（CIP）数据

Azure WebSites 权威指南——微软云计算 Web 平台开发实战详解 / 赵伟编著. —北京：清华大学出版社，2015
ISBN 978-7-302-40014-1

Ⅰ. ①A…　Ⅱ. ①赵…　Ⅲ. ①计算机网络　Ⅳ. ①TP393

中国版本图书馆 CIP 数据核字（2015）第 086741 号

责任编辑：冯志强　战晓雷
封面设计：
责任校对：徐俊伟
责任印制：

出版发行：清华大学出版社
网　　址：http://www.tup.com.cn, http://www.wqbook.com
地　　址：北京清华大学学研大厦 A 座　　**邮　　编：**100084
社 总 机：010-62770175　　**邮　　购：**010-62786544
投稿与读者服务：010-62776969，c-service@tup.tsinghua.edu.cn
质量反馈：010-62772015，zhiliang@tup.tsinghua.edu.cn
印 刷 者：
装 订 者：
经　　销：全国新华书店
开　　本：185mm×260mm　　**印　　张：**21.5　　**字　　数：**534 千字
版　　次：2015 年 6 月第 1 版　　**印　　次：**2015 年 6 月第 1 次印刷
印　　数：
定　　价：　　元

产品编号：061642-01

序

Over the past few years, Cloud technology has taken our industry by storm. Whether you are a high-schooler creating a small site for your sweetheart or a fortune-500 company hosting thousands of resources, the cloud has totally altered the way we do business.

For Microsoft, the birth of Azure in 2010 has been a critical milestone in the development of the company. Azure Websites is one of the most significant parts of Azure in particular as it's uniquely tailored to take advantage of Microsoft's super-successful web server IIS and deliver a scalable solution that can benefit virtually anyone in some way. Features like auto-scale, traffic manager integration and WebJobs are designed from the ground up to meet today's ever-increasing needs.

As a program manager for Azure Websites, having a book about the service dedicated to the Chinese market is something that I've hoped for a while, and I can imagine no one better for this than Wei Zhao. I've worked with Wei for several years, and seeing him do work that affects so many customers is nothing less than astonishing. It takes incredible vision, hard work and perseverance to write and publish a book, and Wei's stamina, intellectual horsepower and attention to detail has made it possible. I'm sure you will greatly enjoy reading this publication, and that it will jump-start your work with Azure Websites into the next level.

Erez Benari
Program Manager
Azure Websites

序

It is very rare that someone gets an opportunity to work on something that they absolutely love. I have spent over decade reshaping Microsoft's web platform in variety of different roles. Early 2012, I was asked to get involved in helping build and run cloud scale high density web hosting platform for Microsoft Azure (Code name "Antares"). There was a clear need in the marketplace to deliver on a service that is fully managed, easy to develop, works across platforms, supports open source, and is highly reliable and secure. Today, Microsoft Azure Websites (Antares) remains the fastest growing PaaS platform in the industry. The team remains committed to innovating and keeping customers happy.

Azure Websites is truly next generation web hosting platform for modern applications. It not only managed the lowest level of stack on behalf of developers and site administrators, but it's highly reliable and available across different data centers around the world. Customers have been using Azure Websites for variety of scenarios like, Organizational Web Presence, Digital Marketing Campaigns, Business Applications, PaaS/SaaS platform and even personal blogs.

I have found following to be the best way to describe Azure Websites to someone,

- Choosing Virtual Machines (IaaS) for Web Hosting is like building your own house.
- Choosing Cloud Services is similar to renting the house.
- Choosing Azure Websites is like living in a luxury hotel! (Everything is managed and catered to your needs.)

The Author, Wei Zhao has played vital role in shaping the past, present and future of Azure Websites. He has been working with early adaptors and customers in emerging markets since day one, providing valuable feedback the engineering team. His subject matter expertise are clearly evident in this book.

It is an honor to be able to write a forward for a book about Azure Websites. Building this platform and helping customers succeed has been the best time of my career so far (hands down)!

Apurva Joshi (AJ)
Senior Program Manager
Microsoft Azure Websites

序

20 世纪 90 年代初，电脑爱好者很容易被识别：他们常常手里揣着一盒软盘，里边存着宝贵的程序及文档；二十年后的今天，几乎人人（包括七八十岁的长者）都放心地把照片和文档存于某些云盘（cloud drive）之中，把程序置于 GitHub 版本控制中，随时随地存取。类似地，那时电子邮件是一种稀有的特权，少数大学或公司有专门的邮件服务器；而过去十年中，大家都有至少一个网络电子邮件账号，可用越来越多的设备收信写信。应用云的提供公司，譬如 Microsoft 和 Google，很多人一步步地把整个办公自动化系统移入云中。21 世纪初，一个朋友靠他制作网页的能力及对文艺的热情，跻身第一批拥有自己博客的人群；而今，人们对发表各种博客或微博习以为常。就这样，“云”进入我们的生活，成为不可缺少的一部分。基于云的各种网络服务使得万维网亲近，便捷，趣味无穷；它们使得信息安全持久，无处不在：我们不再担忧丢失软盘；还可轻松地凭电子门票只带手机去看一场演出。整个世界仿佛运转于一个拥有无穷存储及计算能力的超级计算机上。

自然地，网络服务的开发人员也想利用这无穷的存储空间及机算能力，享用云给予的持久性和遍及性。过去十年，信息界的领头公司们开始针对这种需求，提供给程序员把他们的程序移入云中计算的能力，即“云计算”。云计算作为一个领域和网络服务模式还在起步阶段，而且未定型，共存着许多不同的形式，以符合不同的需求。对于软件开发人员，有最重要的两种形式：设施即服务（IaaS）和平台即服务（PaaS）。设施即服务提供虚拟版的计算设备：虚拟机、虚拟硬盘、虚拟网络设备等。灵活性是它的优点，你可将已有的应用程序直接搬到这些虚拟设备上运行，用以往的方式维护；但这也正是它的缺点，如果你依然需要如以往那样自己管理 OS 补丁，安排软件更新部署方案，监控每台机器的资源利用，根据需求手工安装新的服务器以扩展或缩减你所部署的服务，那你尚未充分认识和发挥云计算的能力。相比之下，平台即服务使程序员集中精力于程序本身，平台提供并维护程序运行的环境，并进一步通过各种高级设施以简化程序员和 IT 管理人员的工作。

Windows Azure 网站是微软云计算家族中的平台即服务式的旗舰产品。尽管它主要应用于网站运营，但它事实上可运行任意的 Windows 程序，并有多种程序语言及框架的支持。Windows Azure 网站的设计主旨为便捷：便于学习，便于软件开发、部署、移植，现有的网站应用可直接部署，已有的编程经验直接适用；便于编程，调试，监控，许多任务可在浏览器中完成；便于扩展，鼠标数击，便可把网络应用缩放，小到少用资源，免费运营，大到占用多台专用服务，甚至遍布全球多个数据中心，动态地给予遍布全球的网站用户流畅高速的体验；便于提供现代网站必需的一些能力，譬如安全性、登录服务等。Windows Azure 网站一方面充分发挥 Windows 及.NET 的优势以成为运营 Windows 及.NET 的最佳平台，但另一方面又与诸多开放代码界的杰出产品集成，借助和发扬它们的特色。从第一天起，Azure 网站即支持 Git/GitHub 代码控制及程序语言，如 PHP、Node.js 和 Python。诸多现成的软件库或包装好的网站程序可直接使用。正因如此，Azure 的开发 portal 上有一个

丰富的网站应用库，Azure 用户可在 portal 上选取并部署完整的网站应用，只要一分钟的时间，Wordpress、wiki 或类似网站便足以运转并对外开放。凭借便捷的使用、丰富的功能以及开放的平台，Windows Azure 网站自问世以来持续快速增长，仅仅面世一年的时间，便有 20 万用户建立了超过 25 万个网站，每月总计 230 亿访问量。它已成为从爱好者开发小网站到大企业开发公共或内部网站的首选平台之一。

虽然我自项目开始便在 Windows Azure 网站开发小组工作，读了本书，我还是更进一步了解了一些使得 Windows Azure 网站与众不同的功能。作为一个不断进化的网络服务，它的新功能高速涌现，令人难以跟上。这本书清晰而全面地介绍 Windows Azure 网站。它一方面是一本详尽的使用手册，仔细讲解如何使用每一个功能；另一方面是一本云计算编程的向导，既提供领域的总览，又介绍云服务编程的最佳实践。

我与作者赵伟因在 Windows Azure 网站组共事而相识。他曾帮助诸多软件工程师和客户解决了许多问题，使得他们的网站成功地在 Azure 上运行。赵伟的经验使得他的书成功融合了程序员的开发体验和平台开发者的内部知识。希望本书的出版能帮助开发者更进一步地发挥 Azure 网站的潜力，能引导开发人员把应用程序移到新兴的云计算平台上，并进而“云优化”这些程序，以达到云端应用所特有的性能和可靠性。

杨喆
计算机科学 博士
Principal Software Engineer Manager, Azure 网站开发组
2015 年 2 月 12 日 于西雅图

前　　言

今天我们所面对的是一个严峻的、令人激动的 IT 时代。信息技术的不断创新推动着各行业的业务创新。企业以前所未有的速度更新着产品形态，服务方式、应用模式和营销策略等等。企业变革的速度与业务创新越来越依赖于 IT 基础结构的敏捷性。传统的 IT 基础架构，受到软件、硬件、资源利用率和流程等多方面的限制，无法快速满足企业在对现有业务流程进行调整或者开展新业务时产生的各类需求。快速成长与转型的现代企业，需要一个动态的 IT 基础结构来支撑，云计算的出现则是构建动态 IT 基础结构的最有效方法。

云计算可以帮助企业从容应对这个时代。2014 年，全球云计算市场规模已经达到 160 亿美元。在国内，云计算作为新兴产业，已经提升到国家战略高度。2015 年 1 月 30 日，国务院印发《关于促进云计算创新发展培育信息产业新业态的意见》，提出要加快发展云计算，打造信息产业新业态，推动传统产业升级和新兴产业成长，培育形成新的增长点，促进国民经济提质增效升级。

微软于 2012 年与上海市政府及世纪互联签署战略合作协议，2013 年正式落地中国并开启公众预览，2014 年成为国际 IT 巨头中首个在华落地的公有云。Microsoft Azure 网站（Microsoft Azure Web Sites/WAWS）是微软云计算平台 Microsoft Azure 一个全新的平台即服务产品。它允许客户使用不同的编程语言（.NET、Node.js、PHP 和 Python 等）开发 Web 站点。WAWS 可以为任何规模的 Web 应用程序提供安全和灵活的开发、部署和扩展选项。使用 WAWS，客户可以充分利用现有的工具开发和部署 Web 应用，而无需管理硬件基础设施和中间件（比如.NET 框架、PHP 等）。微软在 2012 年 6 月开放提供 WAWS 的预览版，在 2013 年 6 月宣布该服务正式上线。WAWS 一经发布便赢得了客户的青睐。截止到 2013 年 12 月底，Azure 网站每月有 11TB 的访问量；120 多个站点日访问量超过百万；有大约 24.7%的付费用户。

2014 年 7 月，微软公司新任 CEO Satya Nadella 提出了“移动为先，云为先”的核心战略，“移动为先、云为先”的策略更多地是从用户体验角度出发，打破微软以往的局限性、封闭性，从用户应用层面打通微软公司所有的产品，并打通所有平台上的用户体验，包括 iOS 和 Android。

2015 年 3 月 24 日，微软公司发布了新的 Azure App Service（Azure 应用服务）。Azure 应用服务使得开发人员可以为任何平台和任何设备开发 Web 以及移动应用，并提供一致的用户体验。Azure 应用服务包含 Azure 网站、Azure 移动服务以及 Azure BizTalk 服务。作为 Azure 应用服务的一部分，Azure 网站被重新命名为 Web Apps（Web 应用）。

尽管仅仅面世几年，但是云计算已经展现出惊人的影响力，并将在未来彻底改造 IT 业。越来越多的企业已经或者正在将关键 IT 系统迁移到云计算平台。向云计算转型已经成为趋势，而这将为 IT 人士带来巨大的机遇。本人有幸在 2011 年开始接触 Azure 网站，参与 Beta 客户的技术支持工作，并负责培训微软技术支持部门的工程师，包括世纪互联工程

师。受到同事 Erez Benari 的启发和鼓励，决定将解决客户实际案例的经验和培训内容系统整理出版。

本书主要介绍微软平台即服务模式的云计算平台产品 Azure 网站（除非另有说明，所有内容完全适用于 Azure 全球环境和 Azure 中国区环境），全书大致分为三个部分。第一部分首先简要介绍了云计算的基本概念并引入微软云计算平台。随后，深入介绍了 Azure 网站的基本架构设计和主要概念。关于 Azure 网站的架构设计对于开发人员和架构师至关重要。这些知识可以帮助开发人员和架构师更好的了解这个产品，从而设计出能够完美运行在 Azure 网站平台的 Web 应用。而这些恰恰是文档中缺乏的信息。

第二部分详细介绍了 Azure 网站的管理、配置和监视。包括如何通过 Azure 管理门户网站管理您的网站，并实时监控网站的运行状况。绑定网站自有域名和安全证书则是客户最疑惑和报告问题最多的部分。这也是第二部分的一个重点。通过总结客户实际问题，本书完整列出了绑定网站自有域名和安全证书的所有情景，并给出了配置实例。另外，本部分还介绍了如何通过编写程序或者脚本来自动管理您的网站。

第三部分占用了本书的绝大多数篇幅，涵盖了从架构、开发、部署和迁移等整个 Web 应用生命周期。阅读完该部分，您可以根据应用的不同，选择合适的工具和方法将您的现有网站迁移到 Azure 网站上来。开发人员和架构师可以了解如何选择最适合的开发框架和部署方式，并通过集成其他 Azure 服务，开发一个高性能、可扩展的基于 Azure 网站的 Web 应用。

如上所述，本书涵盖了 Azure 网站的方方面面，无论您是 IT 管理员、开发人员还是架构师，本书都会帮助您更好的了解 Azure 网站，更自信的转到这个杰出的平台上来。如果您是个人开发者，本书可以节省您无数时间，协助您快速开始开发和部署基于 Azure 网站的 Web 应用。

如果您对于本书有任何的建议，你可以在 waws.cn 上找到我。

致　谢

感谢我的爱人田冰雪。*You Raise Me up* 的歌词恰当地描述了她对于我的意义。

You raise me up, so I can stand on mountains; You raise me up, to walk on stormy seas; You raise me up...To more than I can be.

你鼓舞了我，所以我能站在群山顶端；你鼓舞了我，让我能走过狂风暴雨的海；你鼓舞了我……让我能超越自己。

感谢我的儿子天天，他善解人意、痴迷阅读并且多才多艺。我很享受跟他在一起的每一刻。关于天天，您可以访问 www.drumboy.cn。

感谢同事 AJ，他在 2011 年邀请我加入到 Azure 网站这个杰出的产品。当时这个产品还处于秘密开发阶段，如果没有他的邀请，这本书就不会产生。

感谢同事 Erez Benari，他是一位杰出的业余喜剧演员。正是源于他的激励，我才会有将自己关于 Azure 网站的经验和知识加以总结并与大家分享的念头。

感谢杨喆博士百忙之中阅读初稿并为本书作序。我还要把感谢献给本书的初稿审阅者，美女同事符文波。

最后，我非常感谢 Develop Support for Internet 团队，感谢他们无尽的鼓励和支持。在这个团队，我渡过了开心的十年。您可以在 blogs.msdn.com/asiatech 上找到这些杰出的美女和帅哥们。

目　　录

第 1 章　Microsoft Azure 网站架构

我们正在经历着一个前所未有的变革时代。信息技术的不断创新也推动着各行业的业务创新。越来越多的企业将关键运营机制建立在 IT 基础结构之上，因此，企业变革的速度与业务创新在很大程度上取决于 IT 基础结构的敏捷性。

传统的 IT 基础架构受到软件、硬件、资源利用率和流程等多方面的限制，无法快速满足企业在对现有业务流程进行调整或者开展新业务时产生的各类需求。快速成长与转型的现代企业需要一个动态的 IT 基础结构来支撑，云计算的出现则是构建动态 IT 基础结构的最有效方法。

未来的互联网将会是一个由“云+终端”组成的世界。云计算的出现改变了人们思考 IT 的方式。读者或许认为云计算是一个虚无缥缈的东西，因此本章首先介绍云计算的基本概念以及企业和个人如何受益于云计算，随后简要介绍 Microsoft Azure 云计算平台提供的产品与服务。

Azure 网站是基于 Microsoft Azure 平台的“平台即服务”（PaaS）产品。Azure 网站支持着世界上一些最大的企业的关键 Web 应用，无比稳定。在 1.2 节，详细介绍 Microsoft Azure 网站的架构。最后，对比 Azure 平台三种托管网站的服务，并介绍如何选择合适的服务。

1.1　什么是 Microsoft Azure

1.1.1　云计算简介

云计算（cloud computing）是分布式计算、并行计算、网格计算、网络存储、虚拟化、负载均衡等传统计算机和网络技术发展融合的产物。狭义地讲，云就是拥有大量计算资源的超级数据中心。云计算的核心思想是将超级数据中心的计算资源通过互联网提供给客户。如同水、电等公共资源一样，客户根据业务需要使用计算资源、存储和网络等，根据使用量付费。

云计算是一种服务模式，云计算通过互联网以服务的方式提供动态可伸缩的虚拟化的计算资源。美国标准研究所为云计算技术确定了如下 5 个关键特征：

（1）随需应变的自助式服务。用户可以根据业务需要自主部署计算资源，比如服务器和网络存储资源，并根据业务需要随时调整资源使用量。使用云计算，用户无须因为短暂的尖峰需求而购买大量的硬件。在业务高峰期，客户仅仅需要租用更多的资源，在需求降低时可以随时释放空闲计算资源。客户只需为使用的资源付费。

（2）随时随地的网络访问。用户可以在任何时候，通过各种设备，比如移动电话、平板电脑、笔记本电脑和工作站通过互联网来访问云计算服务。

（3）共享资源池。云计算提供商将大量的物理或者虚拟计算资源汇集起来，通过多租户模型提供给用户使用。用户可以随时将不需要的资源释放到资源池。云计算服务商可以将目前无人使用的资源重新分配给其他用户。系统根据用户的需求动态分配或者重新分配资源来服务于多个用户。

（4）快速弹性扩展。计算资源可以自动地、弹性地根据业务需要增加或者减少。从用户角度来看，资源的数量是没有限制的，可以随时增加任何数量的资源。在不需要的时候，客户可以释放这些资源。

（5）可计量的服务。云计算系统通过检测相关的指标，比如存储、处理能力、带宽和活动用户数等来自动控制，并优化资源的利用。资源的使用可以被监视、控制和报告，资源的使用对于云计算服务提供商和消费者都是透明的。

云计算的典型场景如下：

（1）开关型应用。

开关型应用。如图 1-1 所示，有很多应用只需要在某些时间运行或者在某些时段需要额外的计算资源。比如，有些企业的报表，汇总应用程序只需在月底运行；而在线购物网站可能在旺季需要更多的资源。

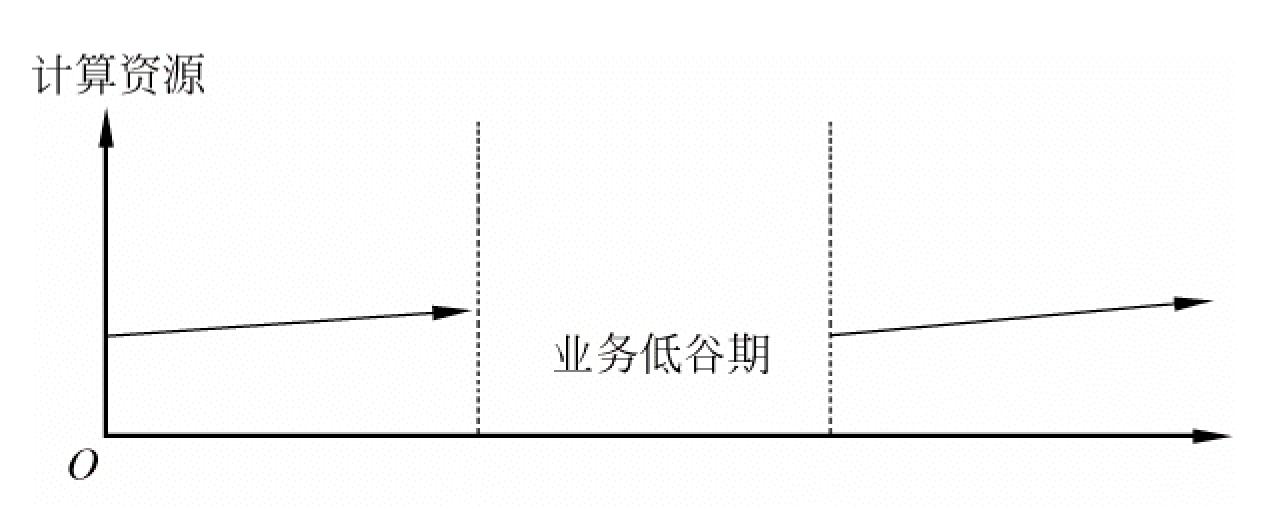


图 1-1　开关型应用

举个具体的例子，某在线购物网站可能在“双 11”期间需要上千台服务器来满足客户的购物需求，可是在购物淡季只需 100 台服务器就可以满足客户需要。如果采用传统的 IT 数据中心，就需要购买上千台服务器硬件以应对购物高峰。但是这些硬件资源在购物淡季闲置，浪费的同时还需要投入人员和资源进行维护。

采用云计算方案，只需在业务高峰期租用更多的资源。根据业务变化，客户可以在淡季随时释放这些资源。客户只需根据使用的资源数量支付费用，降低了成本，并且避免了硬件资源的闲置和浪费。

（2）快速增长模式。

在互联网时代，速度决定一切。很多绝妙的创意如果不能快速推向市场，将会失去意义。如图 1-2 所示，当创新的应用和商业模式被客户认可后，用户基数可能会一夜之间以指数曲线增长，随之而来的是业务的快速增长和相应的软硬件资源需求。

采用传统的数据中心，需要采购更多的硬件，这至少需要几天的时间。如果需要聘请更多的专业人士维护日益庞大的数据中心，可能需要几个月的时间。采用云计算，这一切可以在几分钟内完成。对于企业来讲，云计算的价值不仅仅是降低硬件资源的投资，更关键的是可以帮助客户赢得时间，帮助企业在竞争中存活下来。

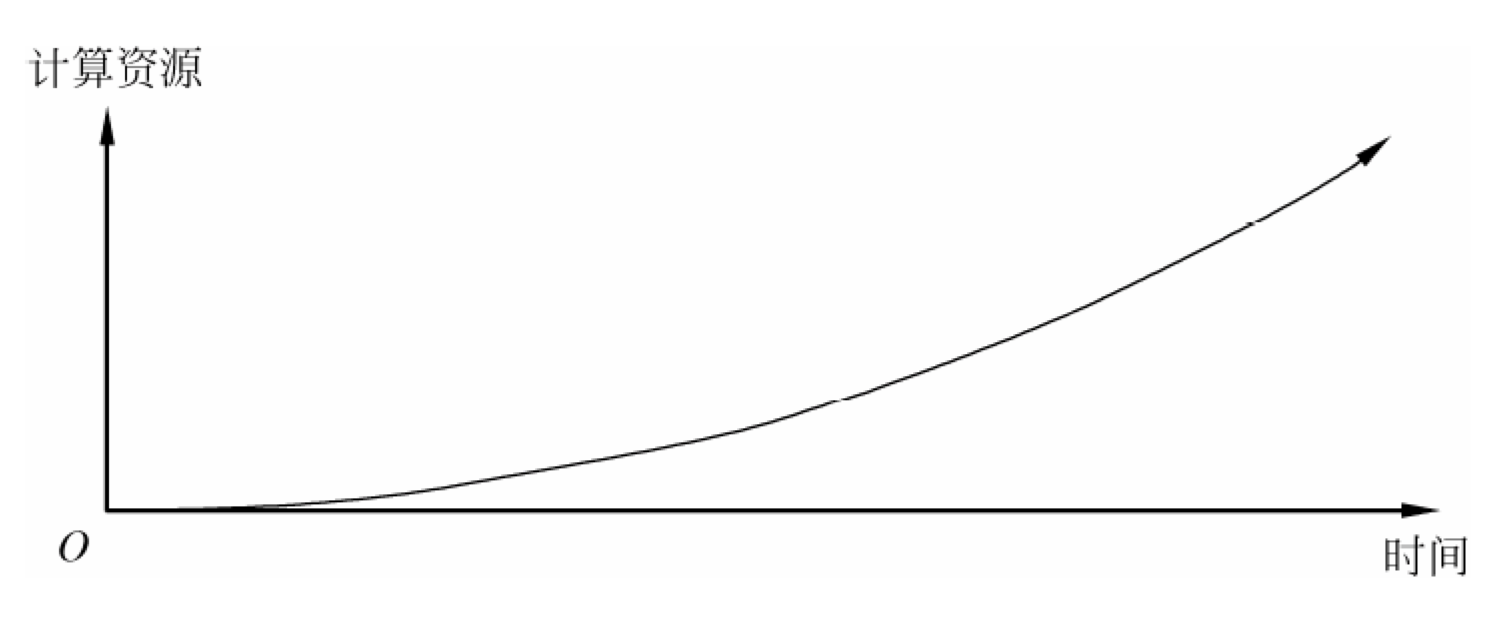


图 1-2　业务快速增长

（3）无法预期的业务高峰。

这种模式可能是所有营销团队的梦想，同时也是 IT 部门的噩梦。如果某个商业推广或者促销活动获得用户的认可，如图 1-3 所示，现有的资源远远不能满足需求。如果 IT 部门不能及时配置足够的资源，可能会导致商业活动的失败；而如果 IT 部门为这种偶发的情况配置足够的资源，将会造成巨大资源的浪费。

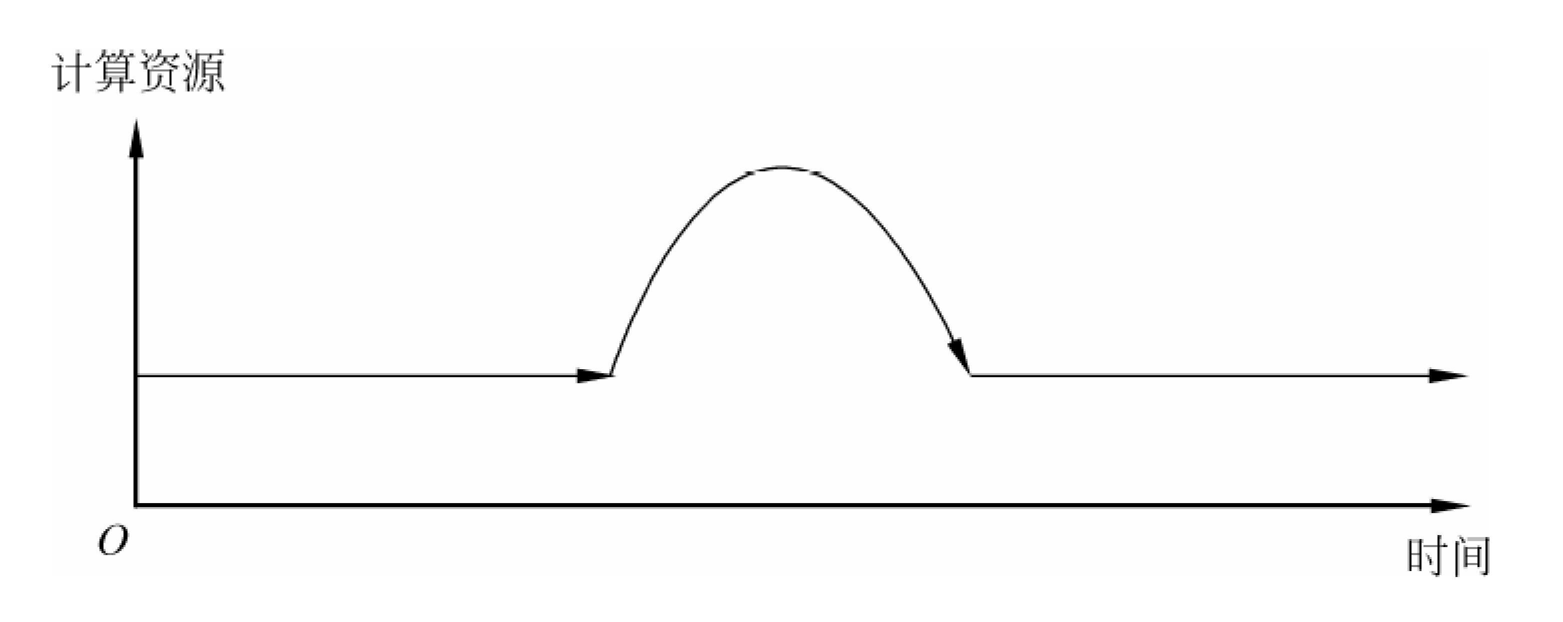


图 1-3　无法预期的业务高峰

采用云计算，不仅可以快速配置足够的资源，同时还可以节省企业投资。

（4）周期性的业务高峰（可预期的业务高峰）。

如图 1-4 所示，云计算可以帮助企业更好地应对周期性的业务高峰。采用传统的数据中心，客户必须按照业务高峰的需求配置资源，从而造成资源过量配置。采用云计算，可以帮助客户降低成本。

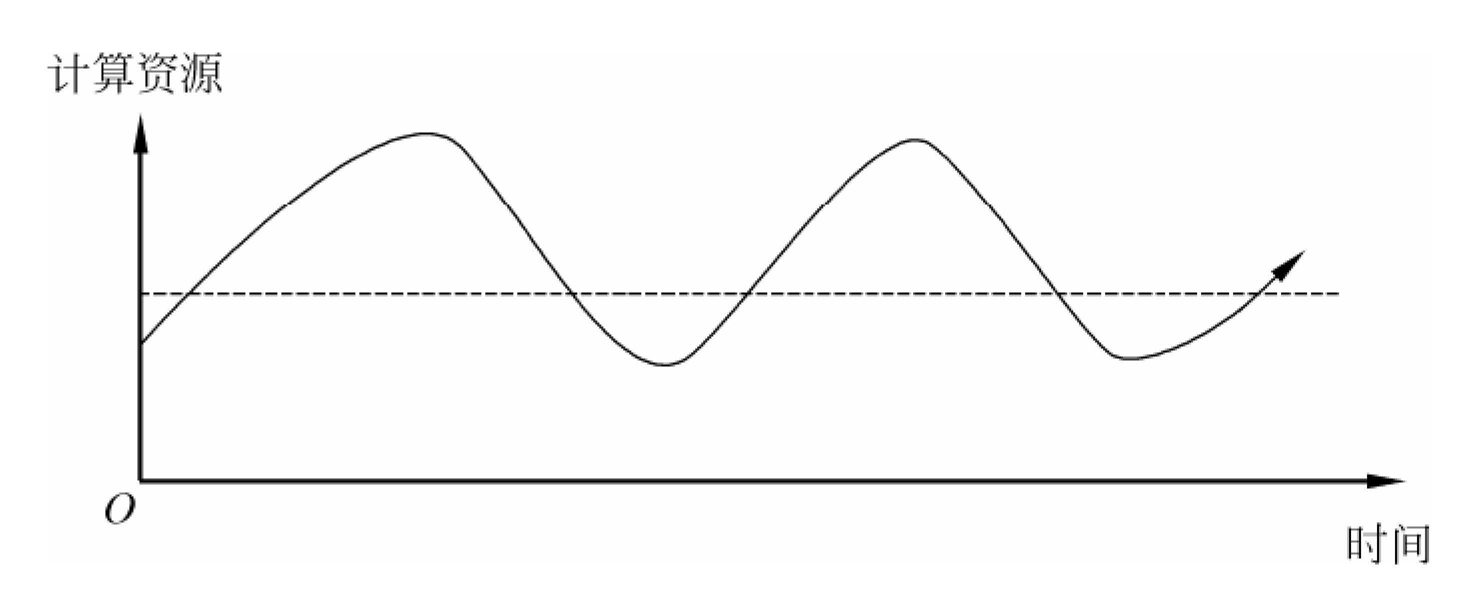


图 1-4　可预期的业务高峰

1.1.1.1　云计算类型

现在，每个企业都在向云计算迁移，至少在考虑向云计算迁移。但是，每个企业的规

模不同，商业需求不同，对于数据安全性的要求也不相同。根据云平台的建设方式和服务对象，通常把云计算分为 3 种类型。

1. 公有云

公有云是指由云服务提供商建设、运营，为外部客户提供服务的云计算平台。对于使用者而言，公有云的最大优点是，其所应用的程序、服务及相关数据都存放在公有云的数据中心，自己无须做相应的投资和建设。

对于中小企业，从成本角度而言，公有云是最佳选择。还有许多公司，比如快速消费品行业、互联网企业、创业公司等，速度就是生命，他们需要快速向市场提供服务，尽快将产品和服务推向用户，公有云是他们的最佳选择。

此外，当产品或者服务获得极大的成功，公司只需租用更多资源即可满足快速增长的用户需求。万一产品失败，随时可以释放计算资源，将损失降低到最小。

2. 私有云

私有云是指企业自己的基础设施，并可以控制在此基础设施上部署应用程序的方式。私有云提供对数据、安全性和服务质量最有效的控制。私有云可部署在企业数据中心的防火墙内，也可以将它们部署在一个安全的主机托管场所。

私有云具备许多公有云环境的优点，例如弹性、快速部署等。两者的差别在于：私有云服务中，数据与程序皆在组织内管理，且与公有云服务不同，不会受到网络频宽、安全问题、法规限制影响；此外，私有云服务让提供者及使用者更能掌控云端基础架构，改善安全性与弹性，因为使用者与网络都受到特殊限制。

对于超大企业，私有云方案可以整合现有资源，提高计算资源利用率，降低运维成本，提高管理效率，提升服务水平，提供商业连续性服务。

相对于公有云，私有云在数据安全方面有很多优势，因为通常私有云构建在企业数据中心，位于防火墙之后。

3. 混合云

混合云是公有云和私有云两种服务方式的结合。基于安全考虑，企业可以选择将关键应用和数据部署到企业内部私有云；同时，将部分非核心业务部署到公有云。很多公有云产品，比如 Microsoft Azure 提供了安全、简便的方法将公有云平台和企业内部数据中心集成。部署在公有云平台的应用可以轻松访问位于企业内部的数据和应用。

1.1.1.2 云计算服务模式

云计算服务提供商通常提供 3 种层次的服务模式：基础设施即服务（IaaS）、平台即服务（PaaS）和软件即服务（SaaS）。

1. IaaS

IaaS 为客户提供基础设施级别的资源服务，比如虚拟机、网络和存储资源等。用户无须购买服务器、网络设备和存储设备，可以从琐碎的基础设施管理与维护中解放出来。IaaS 定位于底层，用户可以在这些资源上自主部署，运行自己的应用。Microsoft Azure 的虚拟机

服务、谷歌的计算引擎（Compute Engine）、亚马逊的 EC2 和 VMWare 的 vCloud 都属于 IaaS。

2. PaaS

PaaS 模式是由云服务提供商提供完整的软件开发和部署平台，通常包括操作系统、编程语言的运行环境、数据库和 Web 服务器。在 PaaS 模式下，开发者不必关心底层基础架构，无须购买和管理底层硬件，也无需购买、安装以及管理中间件，即可进行软件开发。开发者只需关注应用开发本身，从而可以更快速地开发并部署应用。Microsoft Azure 的云服务（Hosted Service）、谷歌的应用引擎（AppEngine）和 Salesforce 都属于 PaaS。

3. SaaS

SaaS 模式由云计算服务提供商在云中管理和运行应用软件，以服务的方式将应用通过互联网提供给用户使用。它是用户获取软件服务的一种新形式。用户不需要购买、安装、管理和运行应用程序。SaaS 有时也被称为“按需使用软件”，云计算服务提供商向用户收取软件的使用费用而不是出售软件给用户。通常按照每用户 • 年或者每用户 • 月来支付费用。微软 Office 365、谷歌企业应用套件等都属于 SaaS。SaaS 的优势是使用简单，初始成本低，无须管理和维护软硬件。

图 1-5 描述了 3 种服务的具体区别。

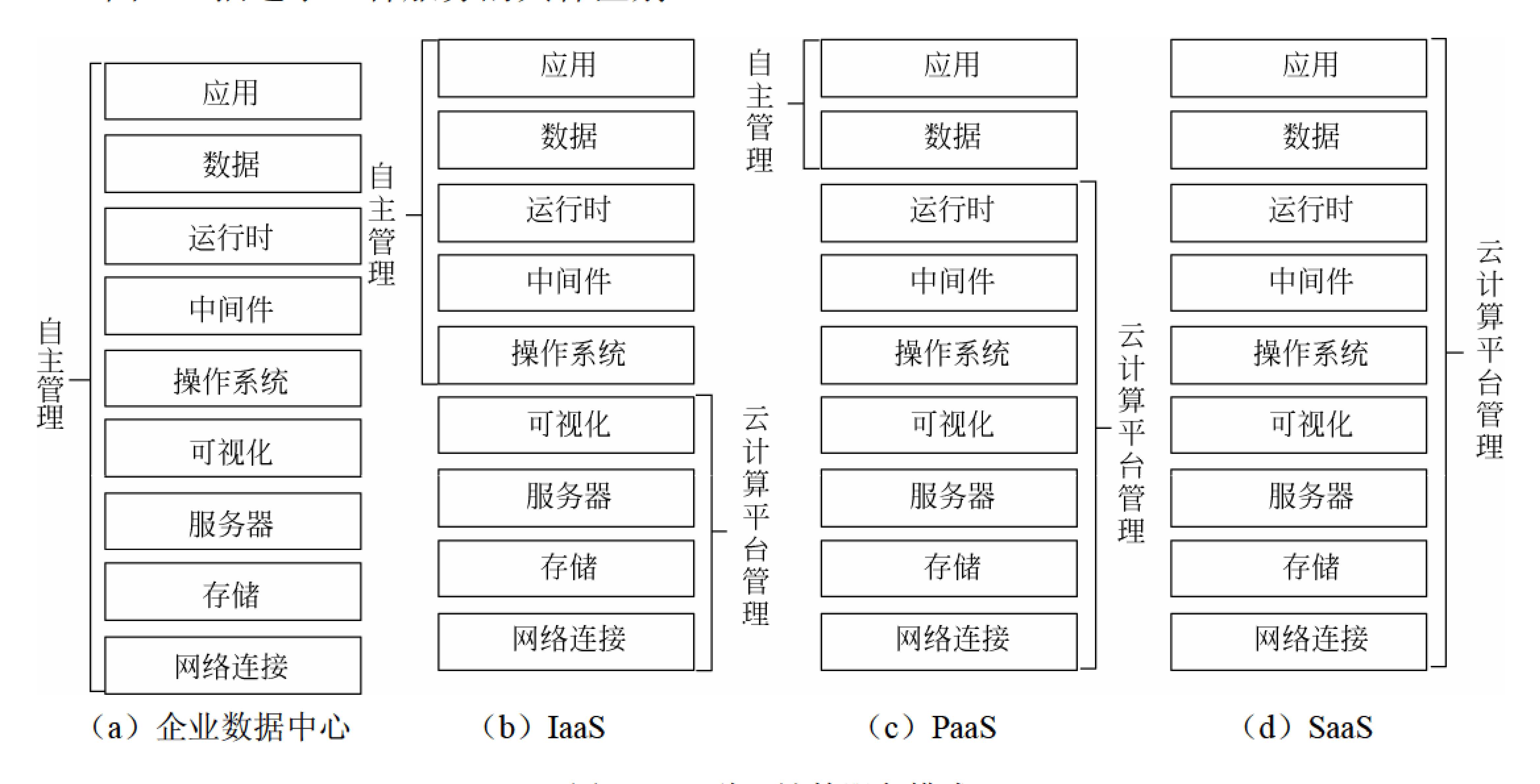


图 1-5　3 种云计算服务模式

3 种云计算模式的根本区别在于服务的灵活性和用户可以控制的范围。在 IaaS 模式下，用户可以最大范围的控制系统，同时由于用户有操作系统的控制权，IaaS 是一个更加灵活的模型。但是，在 IaaS 模式下，用户必须自己管理操作系统，比如安装配置企业应用所需的框架软件、中间件、数据库访问组件和定期安装安全补丁等。

1.1.2　Microsoft Azure 简介

Microsoft Azure 是微软公司运营的一个开放的、灵活的公有云计算平台。通过该平台，

用户可以在微软公司管理的分布在全球的数据中心快速创建、部署和管理应用程序。Microsoft Azure 支持所有主流操作系统、语言或开发工具，并且能够将公有云应用程序与现有 IT 基础设施集成。

微软公司在全球部署了超过 10 个超大规模的数据中心，分布在全球各大洲。在中国微软与世纪互联公司联合建立了 Microsoft Azure 数据中心（位于上海和北京），已经对客户开放，提供服务。

1．Microsoft Azure 的特点

Microsoft Azure 具有如下特点：

（1）永远在线，安全可靠。Microsoft Azure 提供 99.95%的 SLA（Service-Level Agreement，服务水平协议，以月为单位），使用户能够专心开发和运行高度可用的应用程序，而无须花费精力在基础设施建设上。Microsoft Azure 能够自动为操作系统和服务打补丁，并且内置了网络负载平衡和硬件故障恢复功能。Microsoft Azure 云服务与网站服务提供了一种部署模型，支持不停机（zero down time）在线升级用户的应用程序。

（2）灵活开放。Microsoft Azure 支持 Windows 和 Linux 虚拟机。同时，用户可以使用任何语言、框架或工具来构建应用程序。Microsoft Azure 的功能和服务可以使用开放的 REST API 协议访问。Microsoft Azure 的客户端库支持多种编程语言，符合开源许可，并托管在 GitHub 上。

（3）无限扩展。Microsoft Azure 可以帮助用户轻松地将应用程序扩展到任何规模。这是一个完全自动化的自助服务平台，使用户在几分钟之内完成资源配置。用户可以根据业务需要灵活扩展或减少资源使用。用户只需为使用的资源支付费用。Microsoft Azure 的数据中心遍布全球，用户可以将应用部署在靠近客户的数据中心。

（4）功能完备。Microsoft Azure 提供了一个灵活的云平台，可满足任何应用程序需求。用户可以放心托管应用程序，并按需扩展；可以使用 SQL 关系数据库、NoSQL 表存储和非结构化 Blob 存储来存储不同类型的数据，并选择使用 Hadoop 和商业智能服务进行数据挖掘。用户可以利用 Microsoft Azure 健壮的消息传递功能来实现可扩展的分布式应用程序，也可以借助 Microsoft Azure 打造云端和企业本地部署相结合的混合解决方案。Microsoft Azure 中的分布式缓存和 CDN 服务可以降低网络延迟，用户可以从世界任何地方高速访问应用。

2．Microsoft Azure 服务简介

Microsoft Azure 提供的服务如图 1-6 所示。

此外，Microsoft Azure 海报按照类别简要描述了 Azure 提供的服务。访问下面的地址下载 PDF 版本的海报：

```
http://www.microsoft.com/ZH-CN/download/details.aspx?id=35473
```

3．本地开发环境

Microsoft Azure SDK for .NET、Node.js、Java 和 PHP 提供了一些常用工具和资源，可

以用来打包、测试和部署应用程序。Microsoft Azure SDK for .NET 包含 Microsoft Azure Tools for Microsoft Visual Studio，它对 Visual Studio 进行了扩展，能够使用 Visual Studio 方便地在 Microsoft Azure 上创建、生成、打包、运行和调试可扩展的 Web 应用程序和服务。

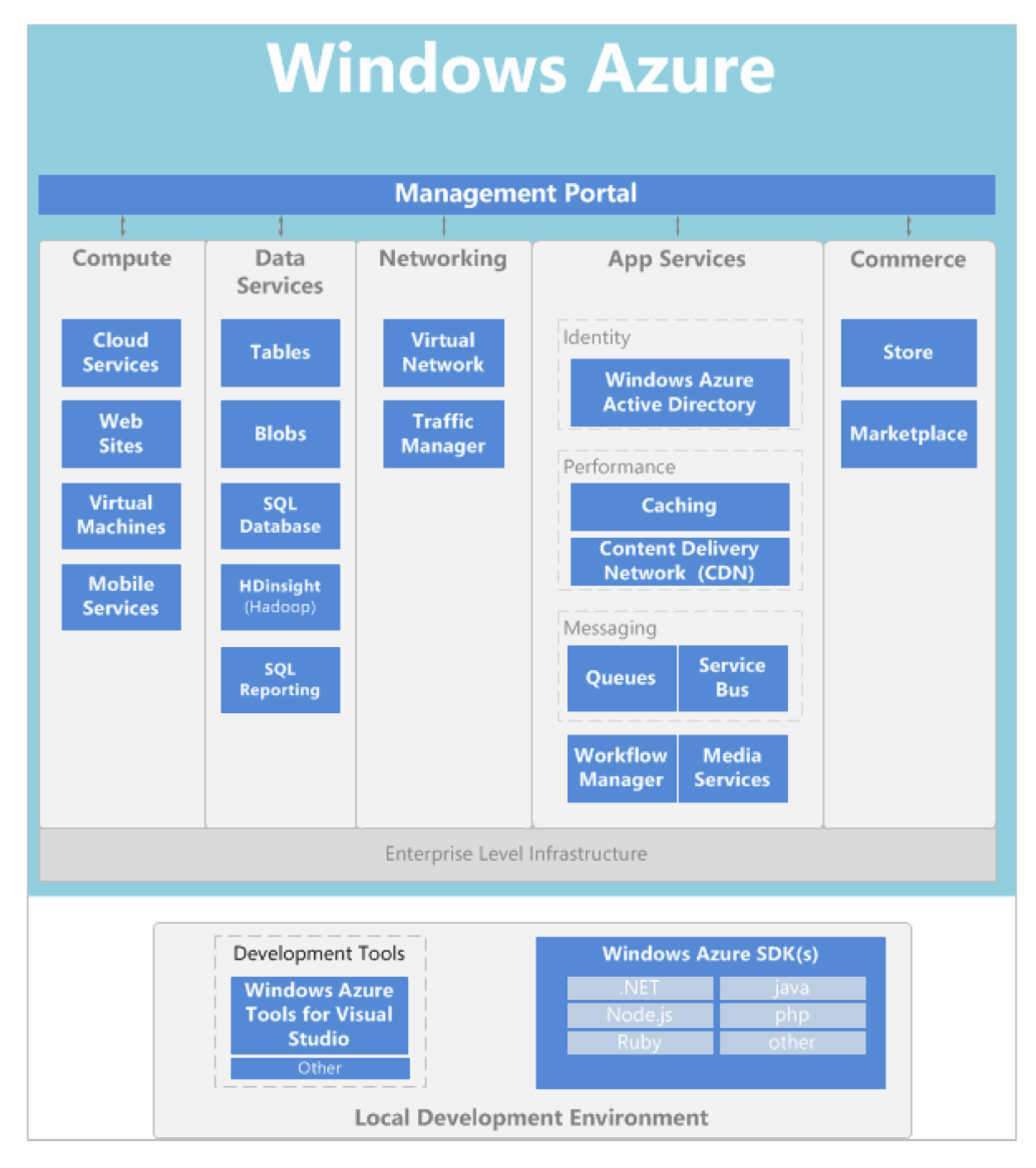


图 1-6　Windows Azure 管理门户功能

4．管理门户网站

Microsoft Azure 管理门户网站是一个 HTML 5 的网站。通过 Microsoft Azure 管理门户网站，用户可以部署、管理各种服务。管理门户网站同时提供服务状态概览信息，用户可以通过管理门户网站了解应用和服务的整体运行状况。管理门户网站对用户的 Microsoft Azure 部署组件加以组织，不间断地提供易于发现和理解的更新信息。可以通过下面的地址访问门户网站。

Microsoft Azure 网站：http://manage.windowsazure.com。

Microsoft Azure 中国网站：http://manage.windowsazure.cn。

1.1.2.1　计算服务

Microsoft Azure 计算服务提供 Microsoft Azure 网站（PaaS）、云服务（PaaS）、虚拟机（IaaS）和移动服务（PaaS）。用户可以根据需要选择适合的功能。

1．网站（Web Sites）

使用任意工具或操作系统，都可以借助 ASP.NET、PHP 或 Node.js 开发并快速部署网站。这也是本书要讨论的主要内容。

2．云服务（Hosted Service）

使用云服务，可以快速部署和管理多层应用程序，Microsoft Azure 将处理部署细节（从设置和负载平衡到运行状况监视）以实现持续可用性。Microsoft Azure 中的云服务由一个应用程序（在云服务中运行）和 XML 配置文件（定义云服务运行方式）组成。

Microsoft Azure 云服务支持两种类型的角色（role）：

（1）Web 角色（Web role）。

是为 IIS 和 ASP.NET 支持的 Web 应用程序编程而自定义的角色。此类型角色的虚拟机默认已经安装 IIS。此角色最适用于为云服务提供基于 Web 的前端。客户通过 HTTP/HTTPS 来访问部署在 Web 角色上的 ASP.NET 应用。Web 角色不适用于长时间运行的进程。

（2）工作者角色（Worker Role）。

是一个对通用开发非常有用的角色，可以为 Web 角色执行后台处理。通常在工作者角色运行服务程序，比如 WCF。Web 角色可以通过 HTTP/HTTPS/TCP 等方式直接访问工作者角色，或者通过 Microsoft Azure 消息队列（Message Queue）与工作者角色进行交互。另外，如果需要执行长时间运行或间歇性任务的后台进程，也应该考虑使用工作者角色。

如图 1-7 所示，一个典型的多层架构的企业应用通常包括前端 Web 服务器、中间层的应用服务器和后端的数据库服务器。

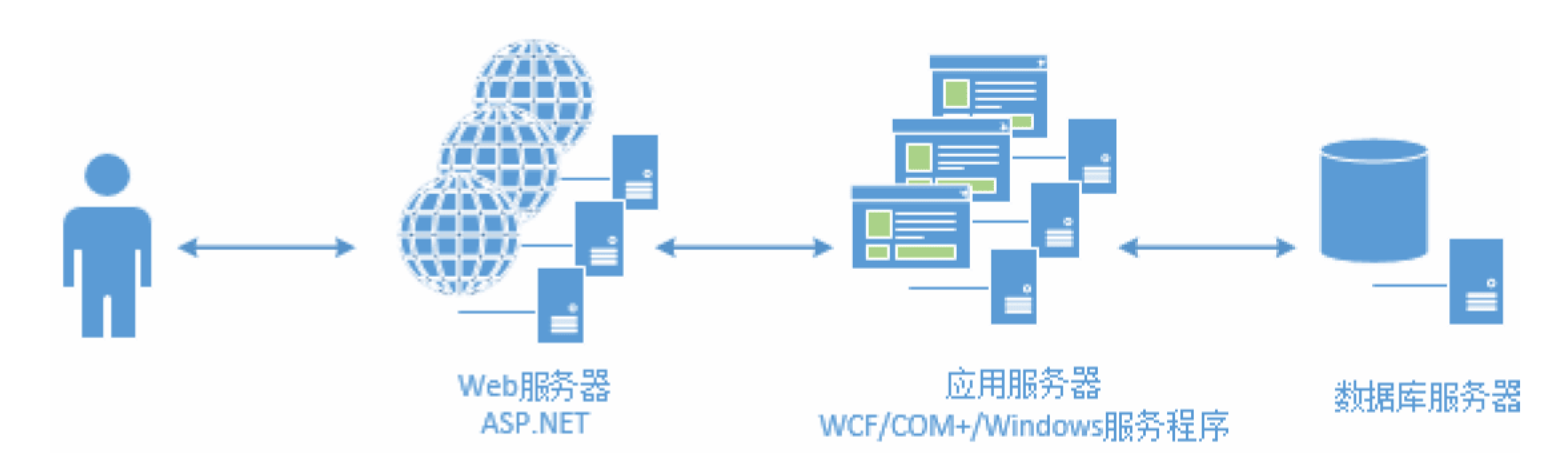


图 1-7　典型多层企业应用

图 1-8 描述了多层架构的企业应用在 Microsoft Azure 的一个典型部署。

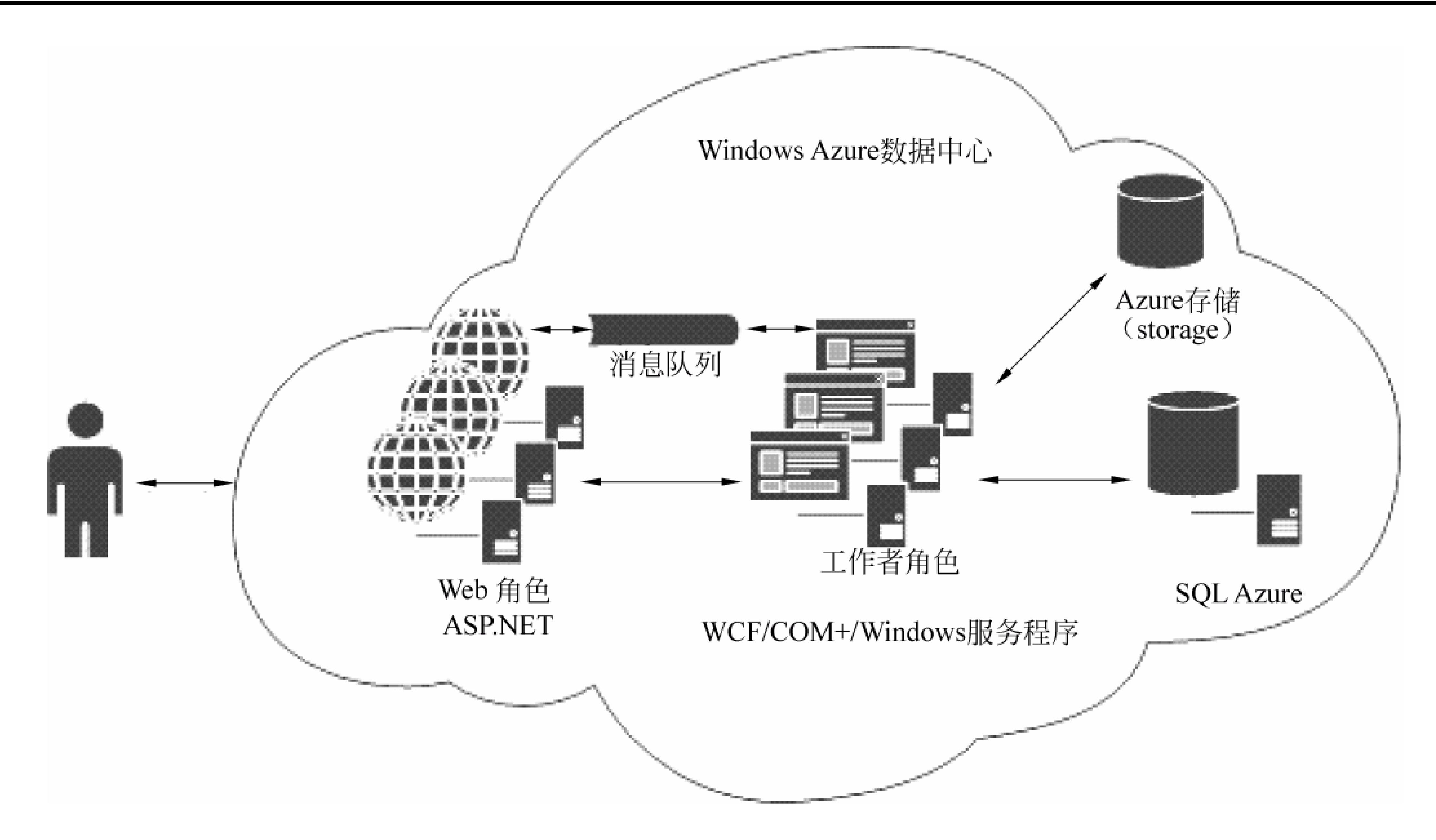


图 1-8　部署在 Azure 上的多层应用架构

3．虚拟机

利用 Microsoft Azure 虚拟机服务，用户可以在 Microsoft Azure 中创建能够完全控制和管理的服务器。在 Microsoft Azure 中创建虚拟机后，用户可以像访问任何其他服务器一样访问该虚拟机，并可以在需要时随时删除并重新创建该虚拟机。Microsoft Azure 虚拟机服务同时支持 Windows Server 和 Linux 操作系统。用户在虚拟机中部署的虚拟硬盘（VHD）可包含自定义设置和用户的应用程序。同时，用户可以创建多个虚拟机，然后对它们进行负载平衡。虚拟机可以与其他 Microsoft Azure 服务无缝集成，比如，虚拟机上部署的应用可以访问 Microsoft Azure 网站、Microsoft Azure 云服务、Microsoft Azure 存储服务和 SQL Azure 等。当然，也可以从任何其他 Microsoft Azure 服务访问虚拟机上运行的应用。

4．移动服务

移动服务为用户的移动应用程序实现后端功能。移动服务的客户端库支持在多种设备（包括 Windows 8、Windows Phone 8、iPhone 和 iPad）上开发移动应用程序。使用 Microsoft Azure 移动服务，开发者可以快速创建和部署能够从任何移动设备访问的高性能的移动应用程序。

1.1.2.2　数据服务

使用数据服务可以在 Microsoft Azure 中存储、修改数据，并生成数据报告。Microsoft Azure 数据服务提供了表（table）、Blob 和 SQL Database 服务，它们可以方便地存储二进制和文本数据、消息、结构化数据以及关系数据。该服务具有众多优点，包括易管理性、高可用性、高可伸缩性和用户熟悉的开发模型。用户还可以使用 SQL 数据同步将关系数据与其他 SQL Database 实例或本地 SQL Server 数据库同步。

Blob 主要用于存储大量的非结构化文本或二进制数据（如视频、音频和图像）。Blob 中存储的数据可以通过 HTTP 或者 HTTPS 访问。BLOB 存储的常见用途包括：

- 通过 HTTP/HTTPS 直接提供影音、图像或文档服务。
- 存储文件的分布式访问。
- 流媒体视频和音频。
- 执行安全备份和灾难恢复。
- 存储数据。

Blob 由 3 个主要组件组成：账号（Account）、容器（Container）和 Blob。图 1-9 是一个典型的 Blob 的示例。下面是一个示例 URL，用来访问 MOV1.AVI：

http://sally.blob.core.windows.net/movies/MOV1.AVI

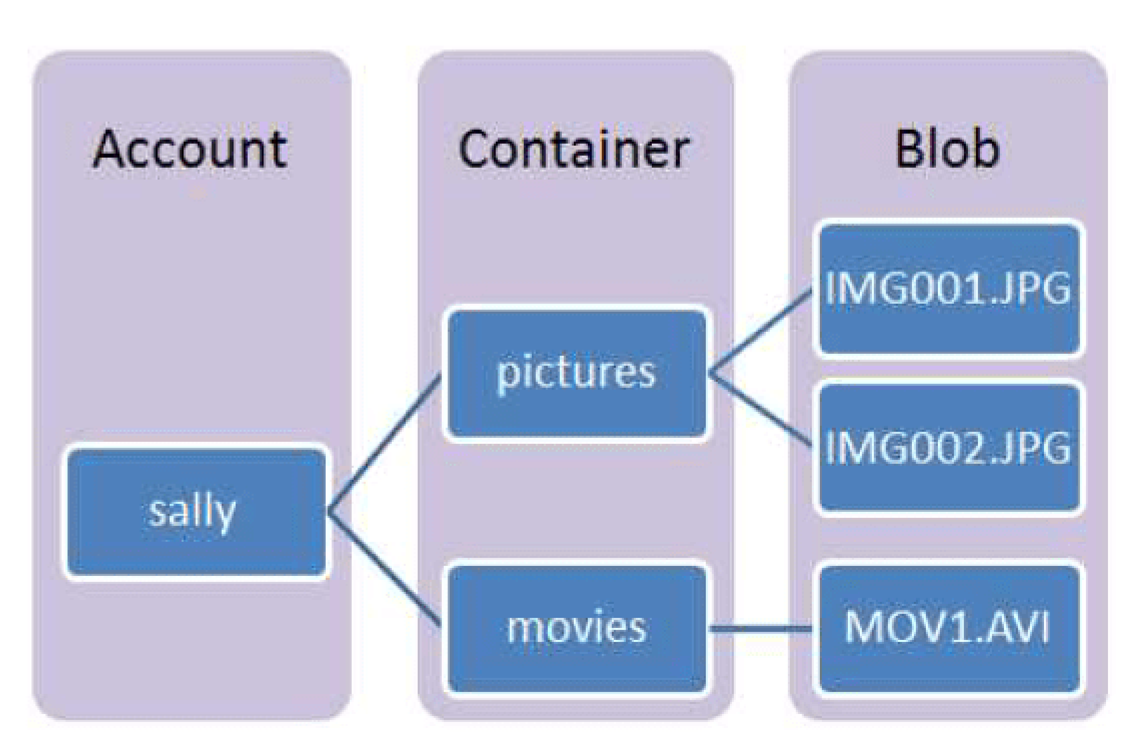


图 1-9　Azure Blob 存储示例

关于如何使用 Blob，请阅读下面的文章：

http://www.windowsazure.com/en-us/documentation/articles/storage-dotnet-how-to-use-blobs-20/

1．表和 Blob

Microsoft Azure 表存储服务主要用于存储大量的结构化数据。该服务是一个 NoSQL 数据存储，可以从 Microsoft Azure 内部和外部访问 Microsoft Azure 表存储服务。Microsoft Azure 的表是存储结构化、非关系型数据的理想选择，表的大小会随着需求的增加而扩展。表存储服务的常见用途包括：

- 为大规模 Web 应用提供结构化数据存储服务。
- 存储无须复杂的连接、外键或存储过程的数据，以便快速访问。
- 使用聚集索引快速查询数据。
- 可以使用 OData 协议、LINQ 查询和 WCF 数据服务访问数据。

表存储服务主要由如下组件组成：

- 账户：Microsoft Azure 存储的所有访问都通过一个存储账户进行。
- 表：是实体的集合。一个单一的表可以包含具有不同属性集的实体。
- 实体：是一组属性，类似于数据库中的一行。
- 属性：是一个名称-值对，比如“电话号码：65432876”。

图 1-10 描述了表存储服务的示例。

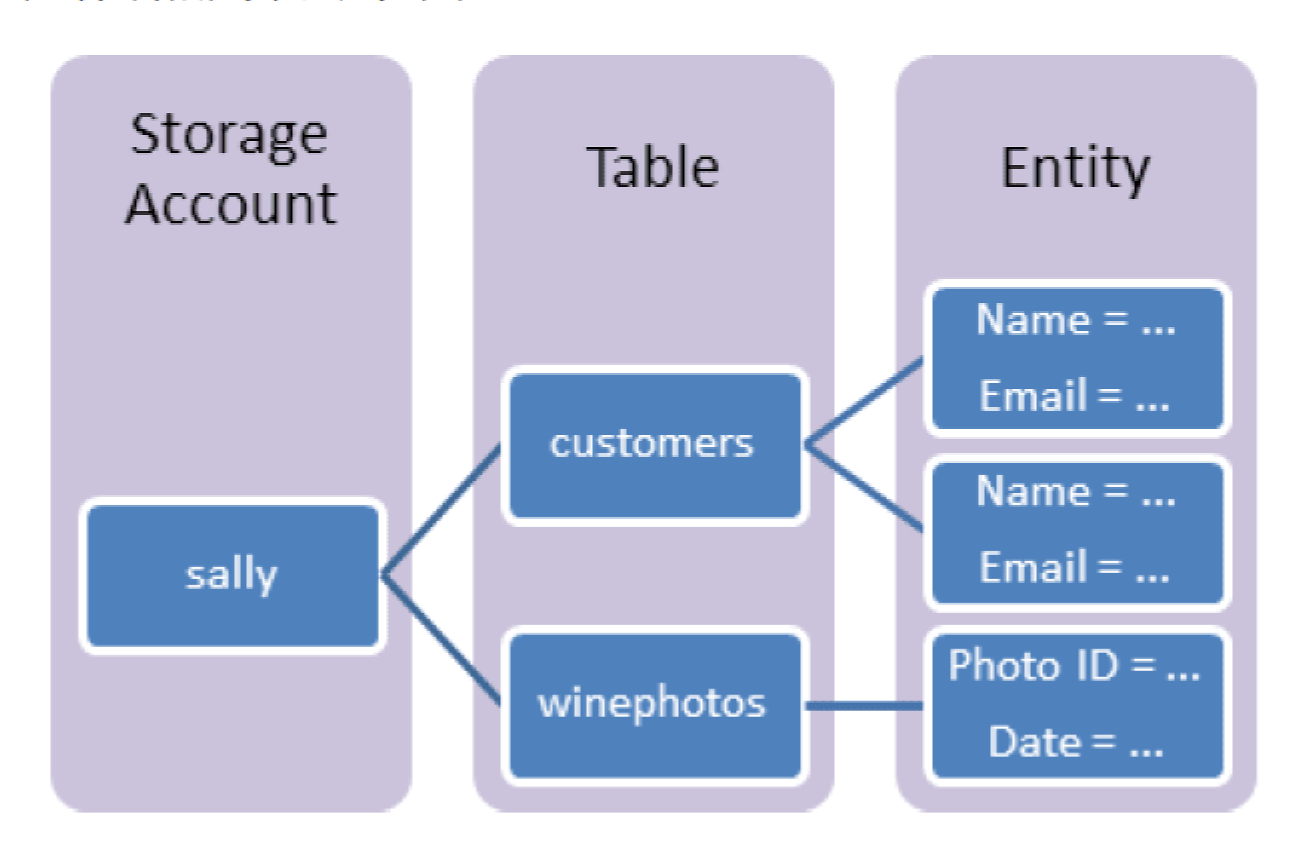


图 1-10　Azure 表存储服务的示例

关于表服务的具体使用方法，请阅读下面的文章：

http://www.windowsazure.com/en-us/documentation/articles/storage-dotnet-how-to-use-table-storage-20/

Table Storage 和 Blob Storage 统称为存储服务，关于存储服务的内部架构，请参考下面的文章：

http://sigops.org/sosp/sosp11/current/2011-Cascais/printable/11-calder.pdf

2．SQL Database

Microsoft Azure SQL Database 是 Microsoft Azure 平台上的关系数据库服务，用于存储大量的关系数据。Microsoft Azure SQL Database 将 SQL Server 的功能扩展到了云环境。使用 Microsoft Azure SQL Database，用户可以轻松地设置和部署关系数据库解决方案。该服务具有众多优点，包括易管理性、高可用性、可扩展性、用户熟悉的开发模型和关系数据模型。

SQL 数据同步可以在 SQL Database 与本地 SQL Server 或其他 SQL Database 示例之间创建和安排定期同步。

SQL Reporting 是基于 SQL Server Reporting Services 技术构建的基于云的报表服务。使用它可以轻松地将报表功能内置于 Microsoft Azure 应用程序中。

HDInsight 基于 Apache Hadoop。它与 Microsoft Office 和 System Center 等工具集成，简化了对大型数据的处理。关于 Microsoft Azure 上的大型数据方案和 Microsoft Azure 上的 Hadoop，请阅读下面的文章：

http://msdn.microsoft.com/zh-cn/magazine/jj190805.aspx

1.1.2.3　网络服务

网络服务在 TCP/IP 和 DNS 级别提供常规连接和路由。

虚拟网络（virtual network）利用 Microsoft Azure 虚拟网络，用户可以将网络扩展到

Microsoft Azure 中。可以在 Microsoft Azure 内设置和管理站点到站点以及点到站点的虚拟专用网络（VPN），并将其与本地 IT 基础结构安全地链接在一起。利用它，还可以根据需要在云环境中将 Microsoft Azure 用于分支机构，或用作受保护虚拟专用网络。

Traffic Manager 使用基于 DNS 的策略，可以控制在同一数据中心或世界各地不同的数据中心运行的应用进行用户流量负载均衡。

1．标识（Identity）

活动目录（Microsoft Azure Active Directory）提供用于在云应用程序中控制和使用标识的多项服务。主要包括：

访问控制服务是一项基于云的服务。通过它，可以方便地对用户进行身份验证，并授权用户访问 Web 应用程序和服务，而在代码中可以不考虑身份验证和授权功能。

Graph API 通过 REST API 端点提供对 Microsoft Azure Active Directory 的编程访问。

身份验证库（AAL）程序开发人员能够轻松利用 Microsoft Azure Active Directory 或其他供程序对用户进行身份验证，然后获取访问令牌，进行安全 API 调用。AAL 还对传入令牌提供验证逻辑，使服务开发人员能够保证其资源安全。

2．性能

缓存服务（Microsoft Azure Cache）利用缓存服务可以轻松地在云中设置缓存，供可能从缓存获益的任意应用程序或服务使用。这包括 ASP.NET 中的会话状态和输出缓存情况。

内容传送网络（Microsoft Azure Content Delivery Network）通过在美国、欧洲、亚洲、澳大利亚和南美洲的超过 20 个的物理节点缓存内容，为开发人员提供了一个高带宽内容传送的全球解决方案。

使用 CDN 缓存 Microsoft Azure 数据有以下优点：

（1）最终用户若远离内容源并且使用的应用程序需要很多“Internet 往返”才能加载内容，则可以通过此功能得到更好的性能和用户体验。

（2）大分布范围，能够更好地处理瞬时高负载，如产品发布等事件开始时。

1.1.2.4 应用程序服务

应用程序服务包括 3 个方面：消息传递、工作流管理和媒体服务。

1．消息传递

1）队列存储（queue storage）

在 Microsoft Azure 中运行的应用程序层之间提供可靠的持续消息传递。它是 Microsoft Azure 存储服务的一部分。Microsoft Azure 的队列存储用于存储大量的信息，可以在世界任何地方通过使用 HTTP 或 HTTPS 访问队列中的消息。在图 1-8 中提到，队列是 Web 角色和工作者角色之间通信的一个重要方式。队列存储的常见用途如下：

- 创建工作任务并进行异步处理。
- 从 Microsoft Azure 的 Web 角色将消息传递到 Microsoft Azure 的工作者角色。

2）服务总线（service bus）

Microsoft Azure 服务总线提供安全且广泛可用的托管基础结构，以实现广泛通信、大规模事件分发、命名和服务发布。服务总线为 Windows Communication Foundation（WCF）和其他服务端点（Endpoint）（包括 REST）提供连接选项。可以将端点置于网络地址转换（NAT）边界的后面并/或绑定到经常更改的、动态分配的 IP 地址。

有关将服务总线集成到应用程序中的更多信息，请参考下面的文档。

http://msdn.microsoft.com/en-us/library/windowsazure/ee732537.aspx

Microsoft Azure 存储队列和服务总线队列各有优势。关于两者的对比和选择，请参见下面的文章：

http://msdn.microsoft.com/zh-CN/library/windowsazure/hh767287.aspx

2. 工作流管理器

工作流管理器提供在大范围、高密度的多租户环境中托管工作流的功能。

3. 媒体服务

媒体服务形成一个可扩展的基于云的平台，使开发人员能够生成可扩展的媒体管理和传递应用程序。

1.1.3　Microsoft Azure 网站简介

Microsoft Azure 网站（Microsoft Azure Web Sites，WAWS）是微软公司云计算平台 Microsoft Azure 的一个全新服务。微软公司在 2012 年 6 月开发提供 WAWS 的预览版，在 2013 年 6 月宣布该服务正式上线。WAWS 一发布便赢得了客户的青睐。截至 2013 年 12 月底，WAWS 每月有 11TB 的访问量，超过 120 个站点的日访问量超过一百万。目前有大约 24.7%的付费用户。

WAWS 是基于 Microsoft Azure PaaS 的一个应用，同时提供 PaaS 模式。它允许用户使用不同的编程语言（.NET、Node.js、PHP 和 Python 等）开发 Web 站点。WAWS 可以为任何规模的 Web 应用程序提供安全和灵活的开发、部署和扩展选项。使用 WAWS，用户可以充分利用现有的工具开发和部署 Web 应用，而无须管理硬件基础设施和中间件（比如.NET 框架、PHP 等）。

相比传统的主机托管，WAWS 具有如下优势。

1. 完全适用于商业站点

- 高可用性，WAWS 提供 99.9%的月度 SLA。
- 24×7 技术支持。
- 高安全性，提供 SNI 和基于 IP 的 SSL 支持。
- 轻松访问其他 Azure 服务，比如缓存、服务总线和存储。
- 遍布全球的数据中心。

2. 无限扩展

- 超过 120 个网站每天有超过一百万的访问量，有些站点超过二百万的日点击量。
- 所有实例默认开启负载均衡。
- 始终最新的基础设施。
- 可以在几秒钟内自动扩展（支持向上扩展和横向扩展）。
- 随时释放不需要的资源，只需支付使用部分的费用。

3. 最好的 Visual Studio 集成

- 轻松移植现有 ASP.NET 网站，只需很少的改动或根本无须改动。
- 在 Visual Studio 中创建、部署、管理和配置 Azure 网站。
- 使用 Visual Studio 远程调试和诊断。
- 可以连接到 TFS 实现团队开发。
- 集成 Visual Studio Online 在线编辑网站。
- 支持后台作业。

4. 快速推向市场

- 在几分钟内创建一个新的 Web 应用程序，无须依赖 IT 人员。
- 用户可以选择使用自己喜欢的工具，比如 WebMatrix、Visual Studio。
- 灵活多样的部署选项，支持 Git、FTP、Web Deploy 和 TFS。
- 支持持续部署，网站更新更快速、容易。
- 支持阶段部署，可以在部署环境和生产环境之间不停机切换。

5. 开放灵活

- 支持 PHP、Node.js、ASP.NET、Python、Java 以及传统的 ASP。
- 支持 SQL Azure、MySQL 和 NoSQL。
- 支持开源的 Web 解决方案、模板和框架，如 Django 和 CakePHP。

1.2 Microsoft Azure 网站架构

图 1-11 描述了 Azure 网站的顶层架构。在每个数据中心都部署有 Azure 网站系统。根据用户数量和数据中心可用机器容量等，在每个数据中心部署有不同数量的部署单元。每个部署单元都是包含完整 Azure 网站功能的独立系统，各个部署单元之间是相互独立的。

同一个 Azure 订阅下的，属于同一个数据中心的所有网站都在一个部署单元中，不会分散到不同的部署单元。比如，Sky 在香港数据中心创建了 10 个网站，那么这 10 个网站都会被创建在一个部署单元中，而不会分散到几个部署单元。

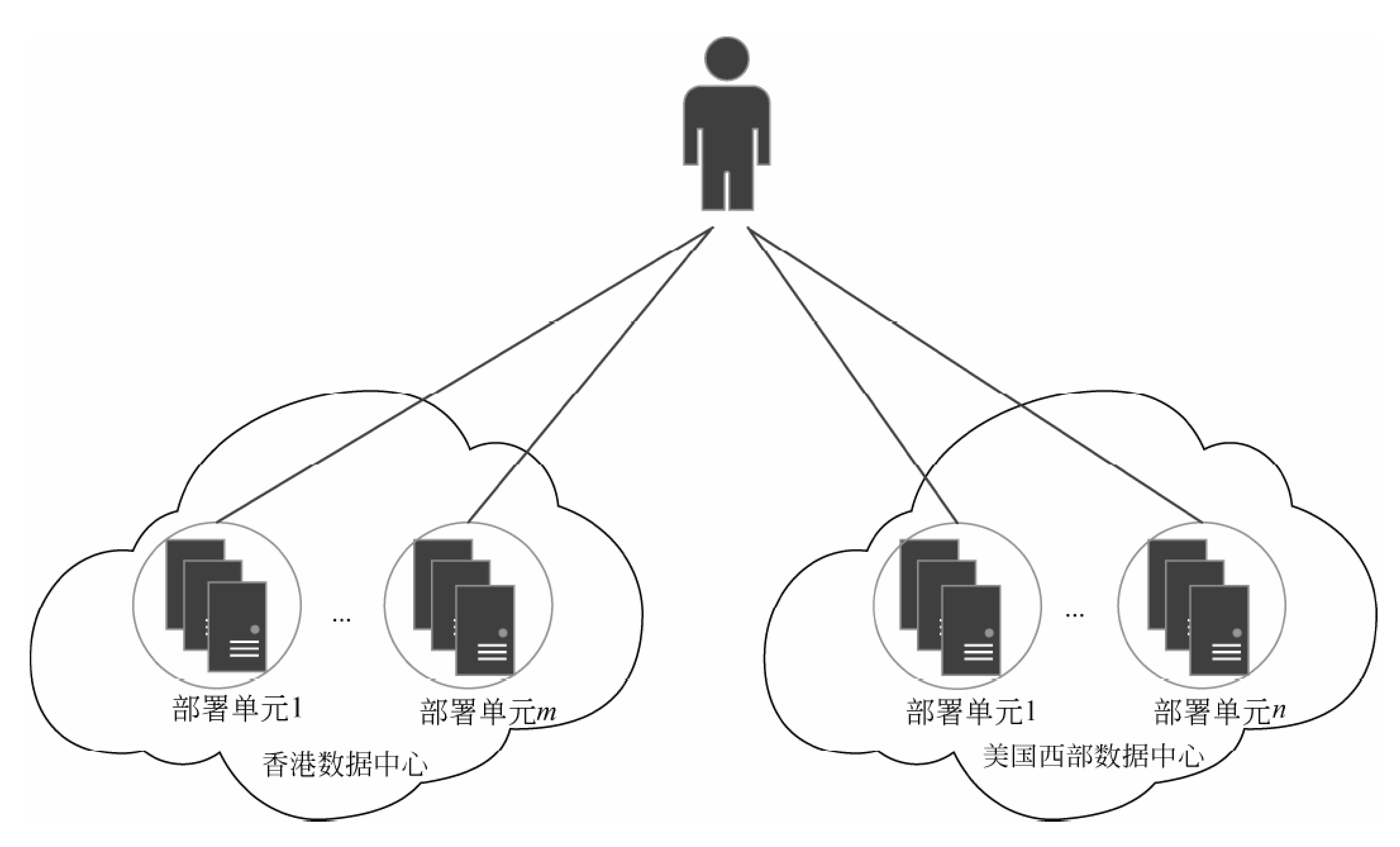


图 1-11　Azure 网站顶层架构

注意：每个 Azure 账户下可以有多个 Azure 订阅（subscription）。订阅可以理解为一个子账户（sub-account）。客户部署的所有的 Azure 资源和使用的 Azure 服务都是登记在某个 Azure 订阅下而不是 Azure 账户下。每个订阅下部署的 Azure 服务被单独计费。企业客户可以根据商业需要创建多个订阅，比如为每个部门创建单独的订阅，便于成本核算。这相当于一个人从同一个运营商处申请了两个手机号码，每个号码被单独计费。

图 1-12 描述了一个 Azure 网站部署单元的基本架构。

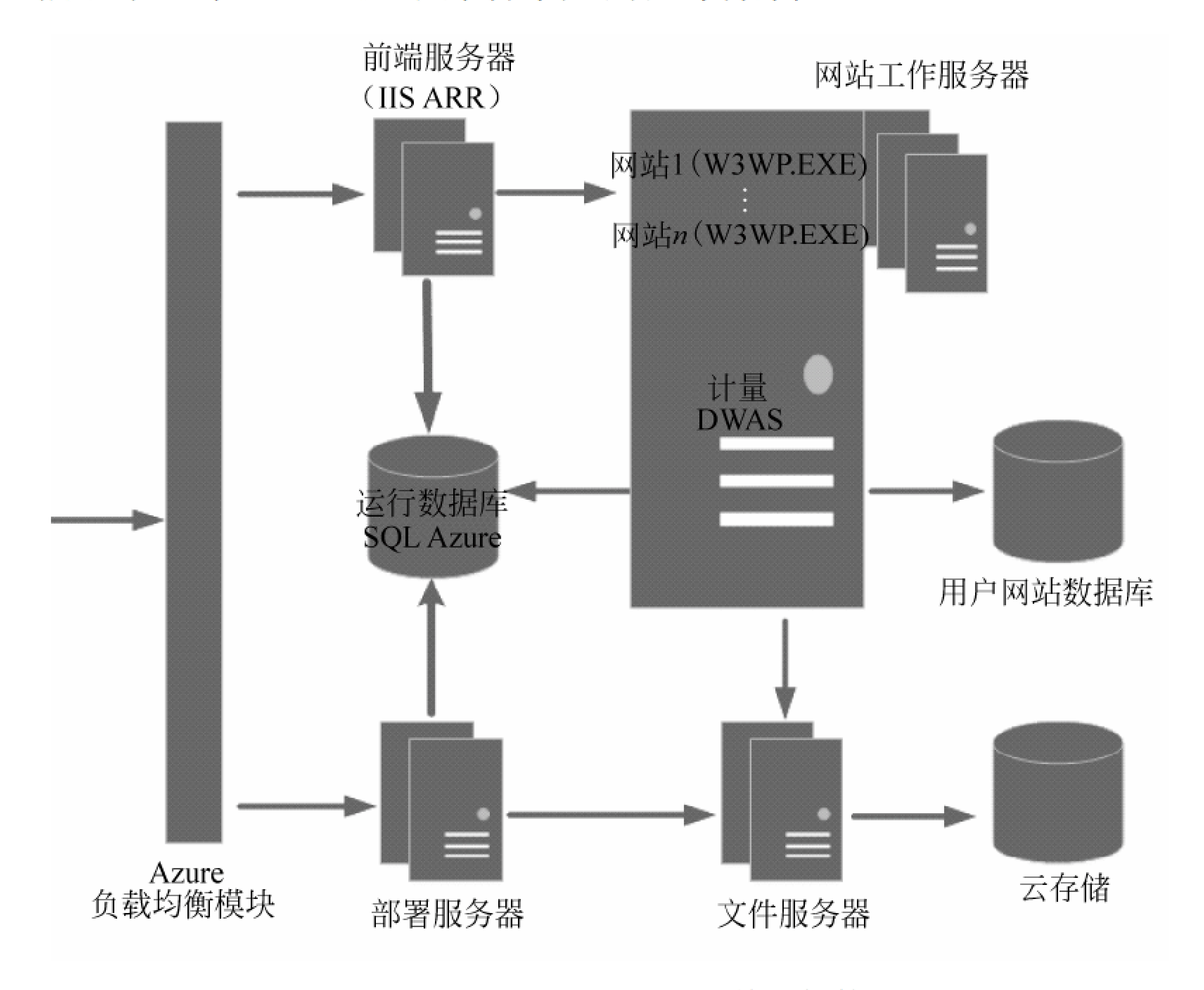


图 1-12　Azure 网站部署单元架构

部署单元由 5 个最基本的核心组件构成。

1. 网站工作服务器（Web worker）

客户的网站运行在网站工作服务器。每个部署单元通常有上千台网站工作服务器。可以简单地认为一台网站工作服务器就是一台 IIS 服务器。每台服务器上可能同时运行很多客户的网站；同样，客户可以指定网站同时运行几个实例，进行负载均衡。工作服务器上安装有运行客户网站必需的语言环境和各种框架，包括：

- IIS。
- .NET 框架。
- PHP。
- Node.js。
- Python。
- Web Deploy/Git。
- 数据访问组件，比如 Microsoft Data Access Component 和 MySQL 的数据驱动程序。

相比传统的 IIS 服务器，Azure 网站工作服务器上运行着两个特殊服务：

1）DWAS

在 Windows Server 2008 操作系统中引入了一个全新的服务：WAS（Windows process Activation Service，Windows 进程激活服务）。WAS 的主要功能是动态创建 IIS 的工作进程（worker process）。在 Azure 网站系统中，对 WAS 服务进行了扩展，称为 DWAS（Dynamic Web Activation Service，动态 Web 激活服务）。与 WAS 服务不同，DWAS 服务的主要工作是动态地注册和销毁网站，而不是动态地创建 IIS 工作进程。

2）计量服务（Metering Service）

计量服务采集工作服务器的资源使用情况，包括 CPU、内存和网络带宽等，并把数据存储到数据库。这些数据用于衡量工作服务器的负载情况，同时用于计费系统。

2. 前端服务器（frontend）

前端服务器是 Azure 网站系统的入口服务器。每个部署单元通常有多台前端服务器。Azure 负载均衡设备负责把客户的请求分发到前端服务器。前端服务器的主要组件是一个工作在应用层（HTTP 层）的负载均衡模块。该模块是基于 IIS ARR（Application Request Routing，应用层请求转发与路由）模块的一个扩展。

前端服务器主要提供两个功能：

（1）负载均衡功能。如果客户网站有多个实例在运行，前端服务器根据配置的负载均衡算法将客户请求分发到各个网站实例。

（2）分配工作服务器。当第一个请求到达时，前端服务器根据工作服务器的负载情况将客户的网站创建在当前负载最低的工作服务器。

同时，前端服务器监视工作服务器的健康状况，自动隔离有问题的工作服务器。

3. 部署服务器（deploy ment server）

Azure 网站提供了丰富灵活的部署方式，比如 Git、Web Deploy 和 FTP 等。部署服务器提供 FTP 部署服务。同时，提供了通过 FTP 下载诊断日志的功能。Git 和 Web Deploy 直接通过客户网站本身完成部署任务。第 5 章详细描述了各种部署方法。

4．文件服务器（file server）

文件服务器通过 Microsoft Azure 云存储提供文件服务，包括客户网站内容文件、配置文件、诊断日志和临时文件等。客户的网站文件保存在文件服务器，工作服务器通过网络共享路径的方式访问网站文件。

5．运行时数据库（runtime database）

运行时数据库存储有客户网站数据和配置等信息，同时保存客户资源使用情况。

1.2.1　请求处理流程

HTTP(S)请求流程（冷启动）如下：

（1）客户访问一个站点（当前并未运行），比如 http://foo.azurewebsites.net。

（2）请求通过 Azure 网络设备到达 Azure 网站的前端服务器。

（3）前端服务器查询运行时数据库，找到当前最空闲的工作服务器。

（4）前端服务器通过工作服务器的 DWAS 服务将 foo 网站建立在第（3）步找到的空闲的工作服务器，并把客户请求转发到该机器。

（5）Foo 网站处理客户请求，将处理结果返回给前端服务器。

（6）前端服务器将处理结果转发给客户端。

HTTP(S)请求流程（访问正在运行网站）如下：

（1）客户访问一个站点，比如 http://foo.azurewebsites.net。

（2）请求通过 Azure 网络设备到达 Azure 网站的前端服务器。

（3）前端服务器保存有 foo 网站信息，直接把请求转给运行 foo 网站的工作服务器。

（4）foo 网站处理客户请求，将处理结果返回给前端服务器。

（5）前端服务器将处理结果转发给客户端。

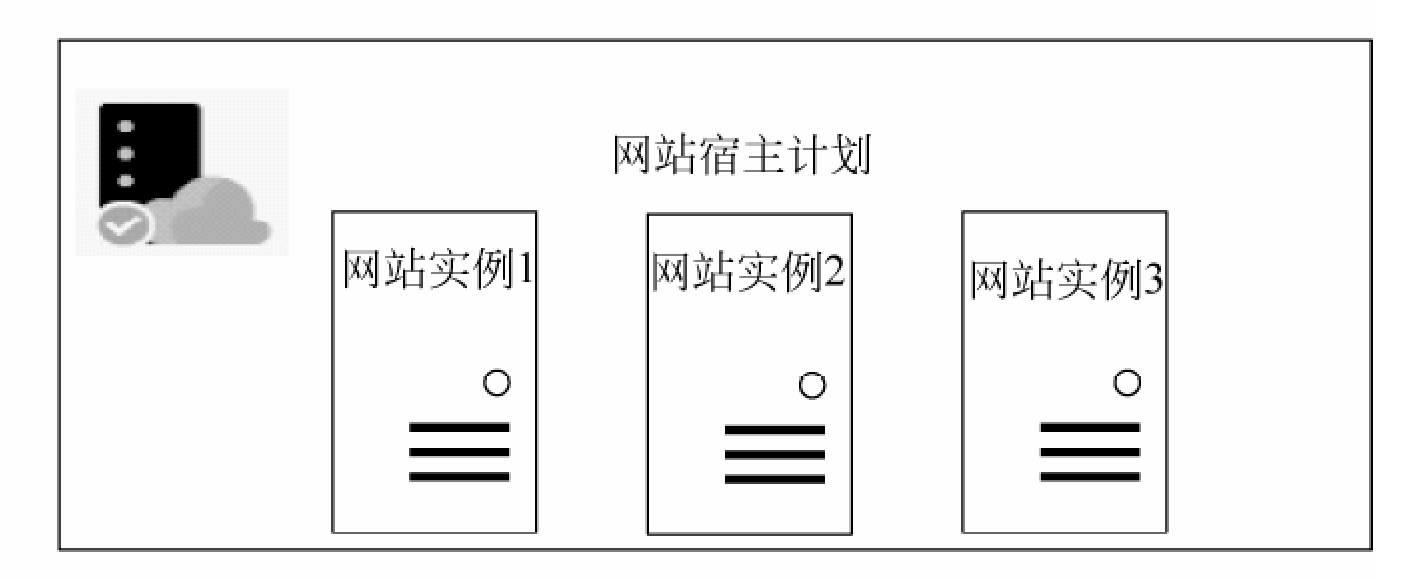
1.3　Microsoft Azure 网站模式

Microsoft Azure 网站提供了 4 个级别的网站方案（免费、共享、基本和标准）。每一级提供了不同的功能和硬件资源。本节主要介绍 Azure 网站划分、网站管理以及每个级别的相应功能。

1.3.1　宿主计划

Microsoft Azure 允许用户通过资源组统一管理某个应用所需的所有资源。资源组是一个逻辑的概念，包含网站宿主计划（Web hosting plans）、数据库服务等。每一个 Azure 网站都属于一个网站宿主计划。图 1-13 描述了资源组的概念。该资源组包含一个数据库和一个网站宿主计划。网站宿主计划包含一个网站，该网站有 3 个实例在运行（运行在 3 台不同的虚拟机上）。





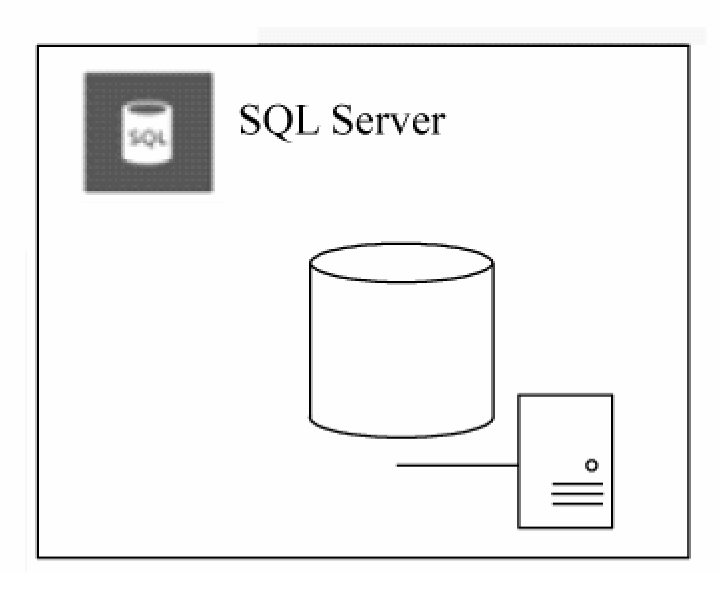


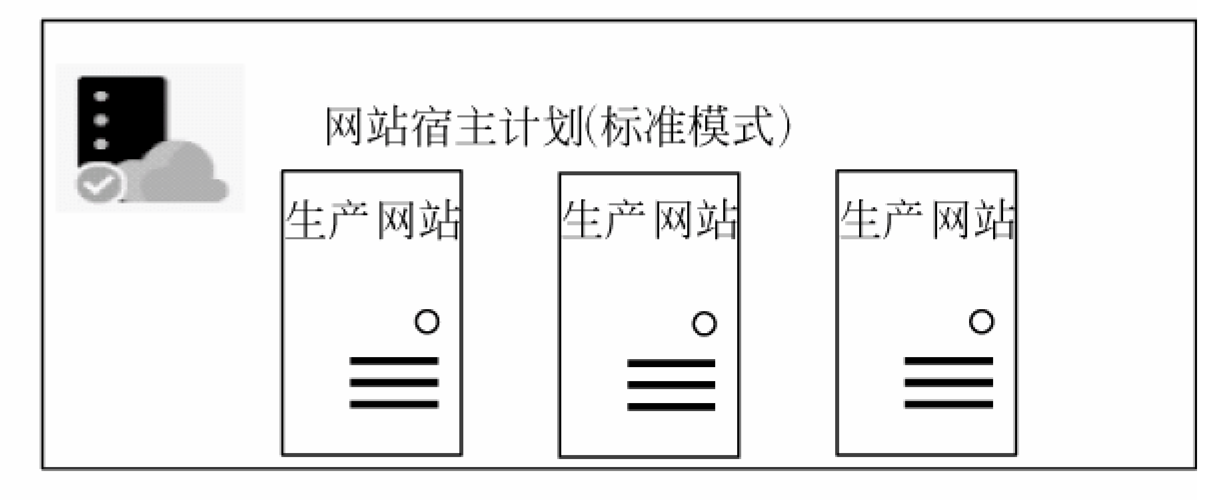
图 1-13 Azure 资源组

网站宿主计划允许用户将网站进行分组，组内的网站共享相同的资源和功能。Azure 支持 4 个层次的网站宿主计划：免费、共享、基本和标准模式，每个层次的网站支持的功能和服务能力都不相同。由弱到强分别为免费、共享、基本和标准模式，免费模式支持的功能最少，标准模式支持的功能最多。

下面通过 3 个例子来详细解释网站宿主计划。

Sky 是网站开发部门的 IT 管理员，他需要同时维护网站的生产环境和测试环境。生产环境的网站需要运行标准模式下，共需要 3 台大型虚拟机。为了节约资源，Sky 希望测试环境运行在基本模式下，使用两台小型虚拟机。如图 1-14 所示，Sky 需要两个网站宿主计划。





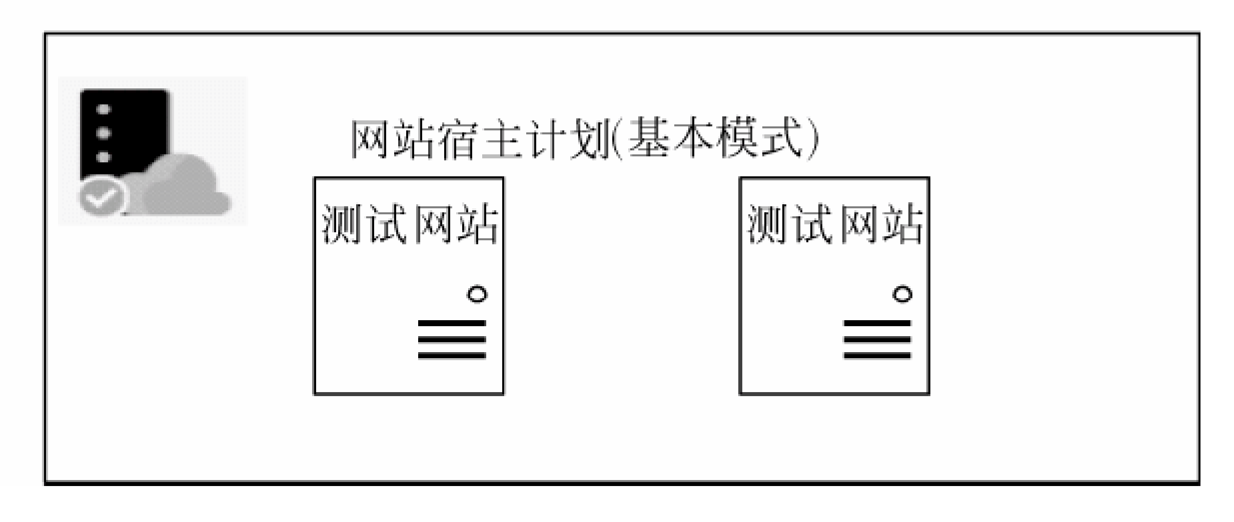


图 1-14 Azure 资源组示例 1

Sky.com 是一个跨国公司，公司的主要客户分布在美洲和欧洲。为了降低网络延迟，提供更好的客户体验，Sky.com 的 IT 管理员将公司网站分别部署到美国东部和欧洲数据中心，如图 1-15 所示。美国站点采用 3 台大型虚拟机，欧洲站点采用两台大型虚拟机。

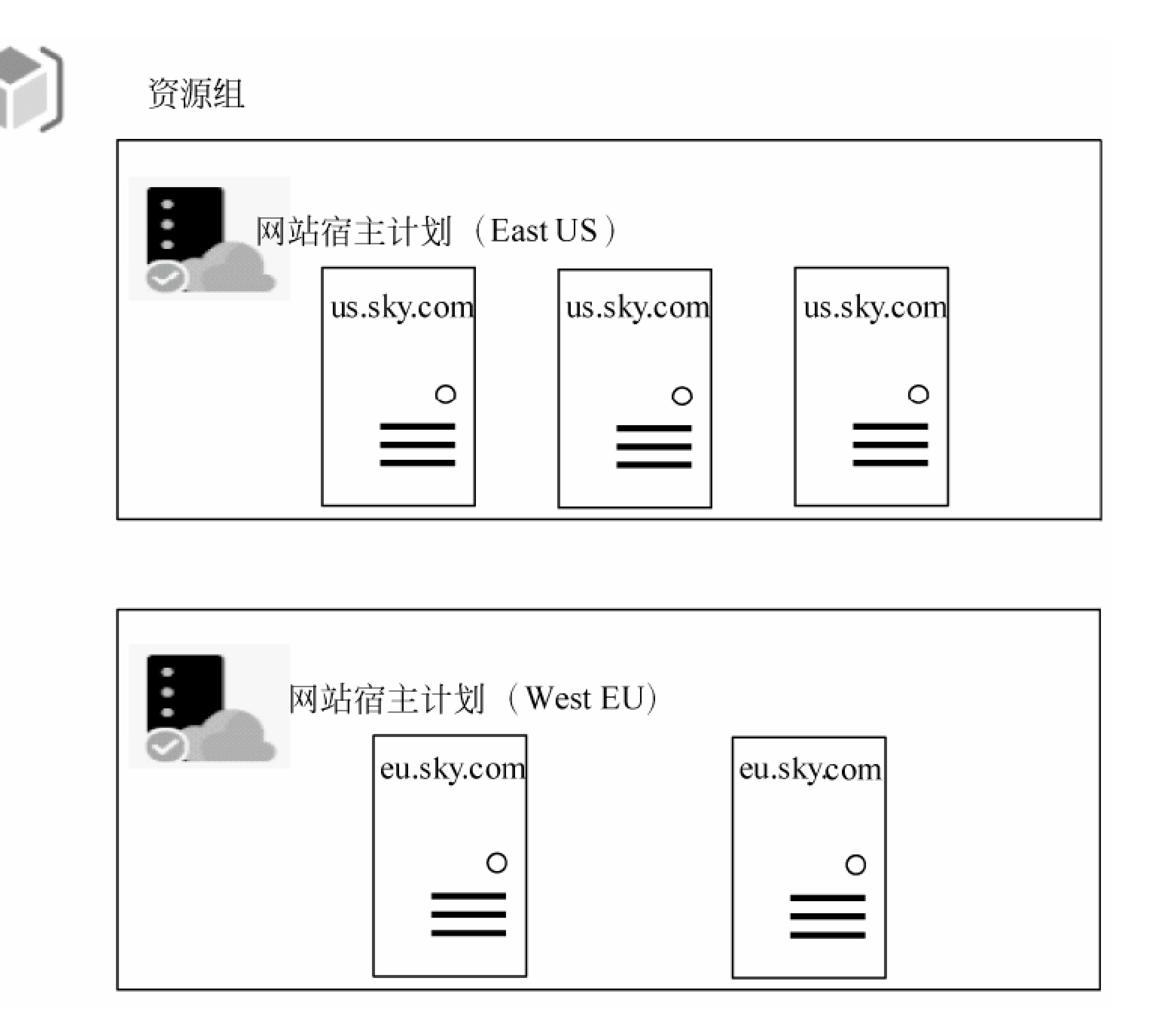


图 1-15　Azure 资源组示例 2

Sky 运行着自己的一个网站托管公司。根据客户应用的需要和支付的费用，该公司将客户划分为 3 个等级：金牌客户、银牌客户和普通客户。如图 1-16 所示，每个金牌客户独享一台大型虚拟机，每 3 个银牌客户共享两台大型虚拟机，每 10 个普通用户共享两台大型虚拟机。

关于站点宿主计划：

（1）属于同一订阅、同一资源组、相同数据中心的网站可以共享一个网站宿主计划。

（2）网站宿主计划只能同时支持一种规格的虚拟机。比如，可以选择使用两台小型虚拟机，但是不能同时选择一台小型虚拟机和一台中型虚拟机。

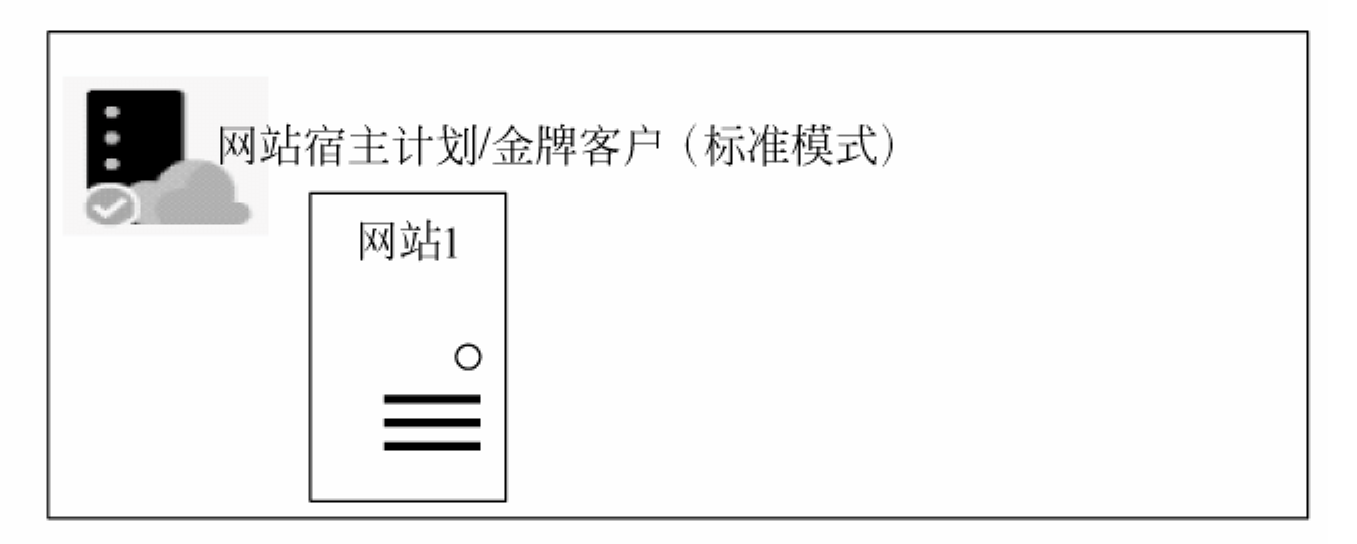
（3）属于同一网站宿主计划的网站共享该计划中定义的功能、特性和资源。例如，有 3 个网站属于一个网站宿主计划，该网站宿主计划配置为使用 3 台中型的虚拟机，如图 1-17 所示，3 个网站将同时运行在 3 台虚拟机上，而不是每个网站一台虚拟机。如前所述，前端服务器提供负载均衡功能。

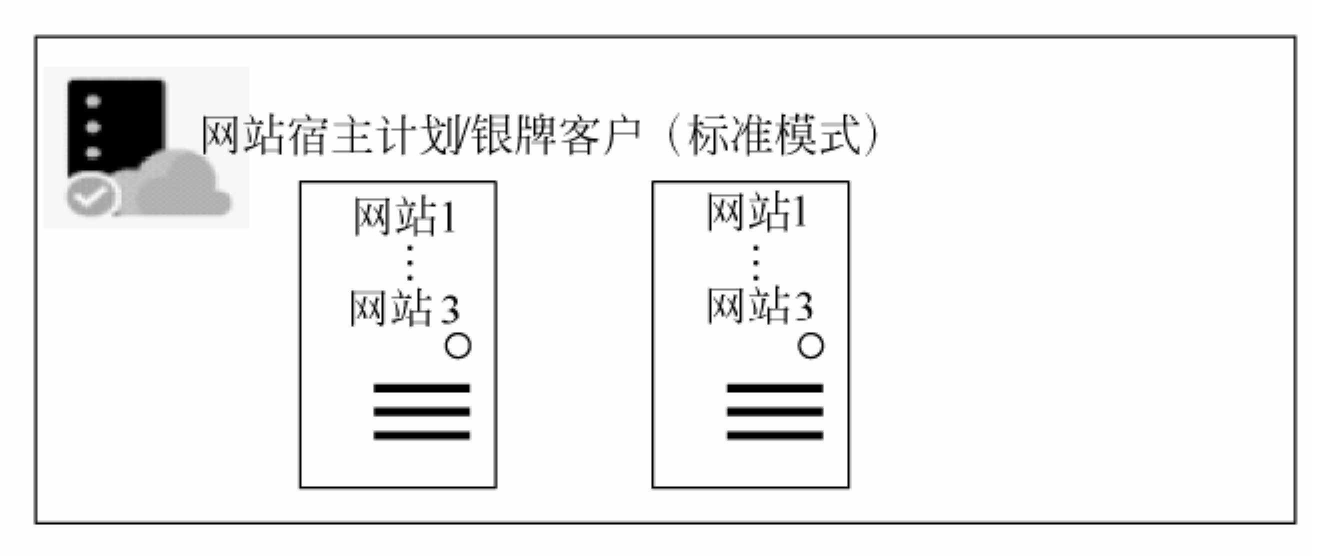
（4）一个网站在任意时刻只能属于一个网站宿主计划。可以将一个网站从一网站个宿主计划转移到同一数据中心的另外一个网站宿主计划中。

（5）网站的扩展以网站宿主计划为单位，如果将某个网站宿主计划从免费模式升级为标准模式，那么属于该计划的所有网站都被升级为标准模式。

（6）目前不能在管理门户网站中直接创建网站宿主计划，只能在创建新网站时同时创建新的网站宿主计划。







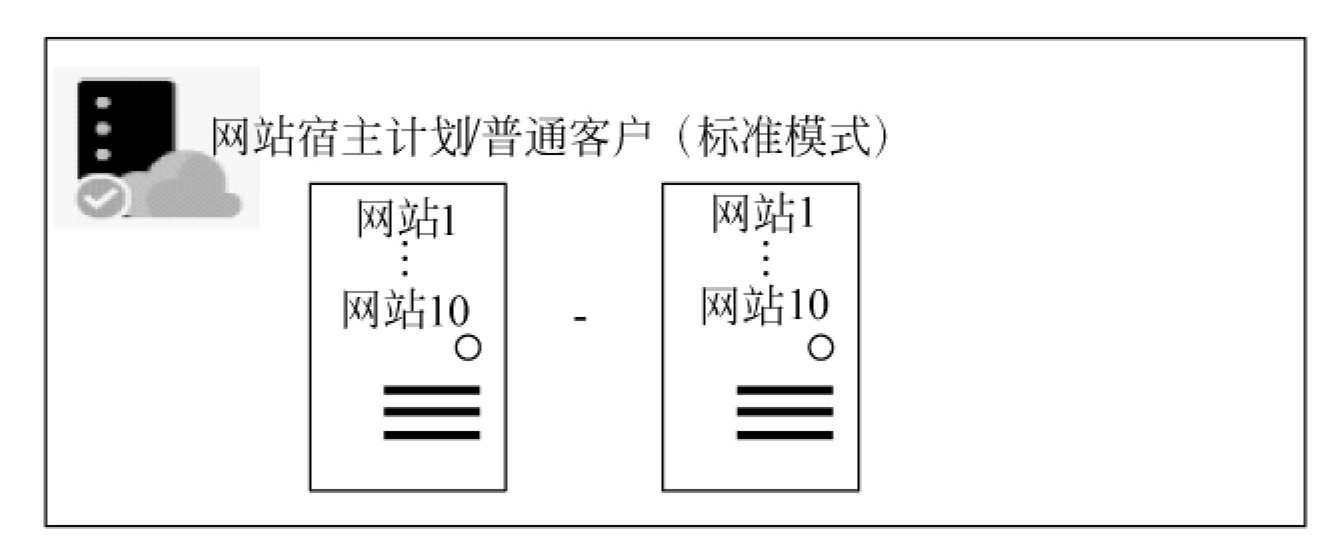


图 1-16 Azure 资源组示例 3

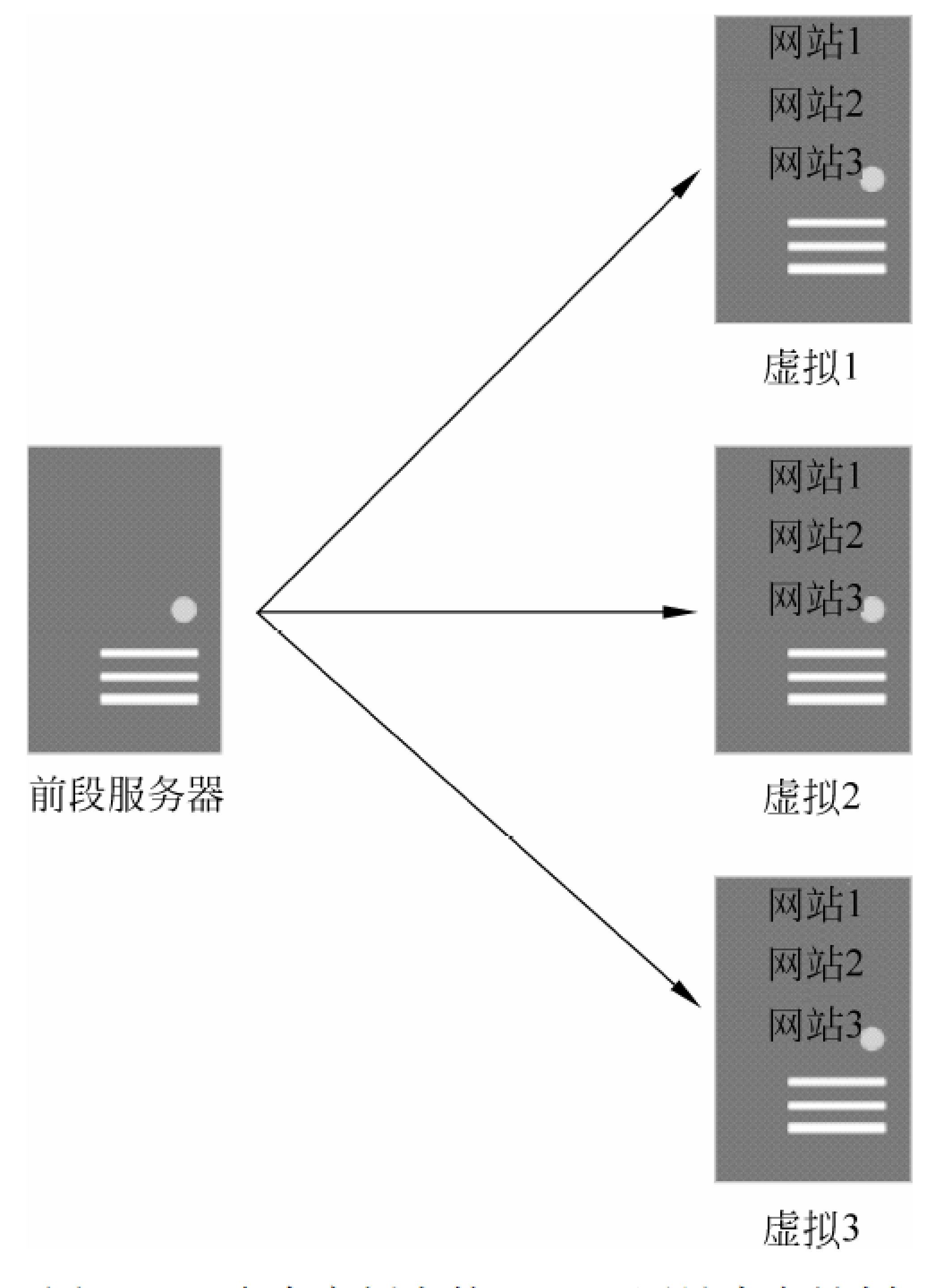


图 1-17 多个实例上的 Azure 网站宿主计划

1.3.2　Azure 网站宿主计划模式详解

网站宿主计划提供了 4 个层次的站点模式：免费、共享、基本和标准。免费模式和共享模式不提供 SLA，基本模式和标准模式提供 99.9%的 SLA。

1.3.2.1　免费模式

在免费模式下运行的网站提供的资源配额和性能比较有限，建议采用免费模式网站进行开发任务或概念验证工作（proof of concept）。每个用户在每个数据中心可以创建 10 个免费网站。在同一个数据中心的免费模式网站共享资源配额限制，如 CPU 和磁盘空间。比如，一个免费模式网站的 CPU 时间超出配额，在同一个数据中心的所有免费模式网站都将被挂起，直到下一个配额间隔周期才会再次启动。如果某个免费模式的网站使用了大量的磁盘空间，那么所有的网站都会受到影响。

如果将生产网站（提供正式商业服务的网站）配置为免费模式，建议在正式上线之前估算负载，避免网站因为使用超过配额限制的资源被停止。

1.3.2.2　共享模式

共享模式网站是一个低成本的可扩展模式，可提供高可用性和与免费模式相比更好的性能。用户可以很容易地通过管理门户将网站设置为共享模式。只需要几秒钟即可完成设置修改，并且不需要改变代码或重新部署应用程序。每个数据中心最多可以创建 100 个共享模式网站。

在同一个数据中心运行的共享模式网站共享一定数量的免费资源配额。与免费模式不同的是，在超出配额后，共享模式网站不会被停止，但需要支付超出配额的资源使用费用。

共享模式允许客户绑定自有 DNS 域名，DNS 配置支持 CNAME 和 A 记录。共享模式网站支持多达 6 个实例（ 横向扩展）以及更高的性能和容错能力。如前所述，WAWS 自动为多个实例提供负载均衡。

在免费模式和共享模式中，多个用户的网站部署在同一台虚拟机上。如果用户需要独享一台虚拟机，那么需要采用基本或者标准模式。

1.3.2.3　基本模式

在基本模式下，用户的网站独享一台虚拟机。相比免费模式和共享模式，基本模式网站将提供高可用性和更稳定的性能。用户可以拥有无限制的基本模式的网站。

基本模式最多支持 3 个实例（横向扩展）。除了免费模式和共享模式的所有功能外，基本模式还支持永远在线（always on）功能。同时，每个基本模式的站点支持 350 个并发的 Web Sockets 连接。

Azure 网站基本模式提供了 3 种规格的虚拟机（小型、中型和大型）。

1.3.2.4　标准模式

在标准模式下，用户的网站独享一台虚拟机。相比基本模式，标准模式网站还支持自

动扩展（auto scale）、站点备份（backup）、阶段部署等功能。标准模式的网站没有数量限制。

1.3.2.5 基本模式和标准模式价格比较

Azure 网站基本模式和标准模式提供了 3 种规格的虚拟机（小型、中型和大型）。表 1-1 列出了 3 种规格的虚拟机配置及价格。

表 1-1 Azure 网站价格表

规格	CPU	内存/GB	基本模式	标准模式
小型	1	1.75	0.075 美元/小时（约 56 美元/月）	0.10 美元（约 75 美元/月）
中型	2	3.5	0.15 美元/小时（约 112 美元/月）	0.20 美元（约 149 美元/月）
大型	3	7	0.30 美元/小时（约 224 美元/月）	0.40 美元（约 298 美元/月）

在同一个数据中心，一个标准模式的网站宿主计划可以扩展到多达 10 台虚拟机。同时，WAWS 支持自动扩展。比如，可以设定在当前的机器 CPU 使用率达到 80%后自动增加一台虚拟机。WAWS 自动为多台虚拟机提供负载均衡。

1.3.2.6 4 种模式的功能对比

表 1-2 列出了 4 种模式所支持的功能。

表 1-2 Azure 网站 4 种模式功能对比

功能	免费模式	共享模式	基本模式	标准模式
站点数量	10 个	100 个	无限制	无限制
存储空间/GB	1	1		10
虚拟机实例	共享	共享	专属	专属
Web 作业	✓	✓	✓	✓
自有域名绑定		✓	✓	✓
横向扩展		最多 6 个实例	最多 3 个实例	最多 10 个实例
自动扩展				✓
负载均衡		✓	✓	✓
自有域名 SSL 支持			✓	✓
始终可用			✓	✓
Web Sockets			350 个	无限制
Azure 调度服务				✓
阶段部署				✓
自我修复				✓
备份与恢复				✓
SLA			99.9%	99.9%

注：
① 以上价格不适用于中国大陆地区。
② 存储空间限额以数据中心为单位，由属于该数据中心的所有网站共享。

1.4　如何选择 Azure 服务

Microsoft Azure 中提供了几种服务来运行用户的 Web 应用程序，如 Microsoft Azure 虚拟机、云服务和网站。这 3 种服务都允许用户运行高度可扩展的 Web 应用程序，但是每个选项都有各自的特点，用户可能无法确定哪一个最适合用户的需求，或者用户可能不清楚相关的概念。下面的内容将帮助用户根据应用的需要选择正确的 Azure 服务。图 1-18 来自 Microsoft Azure Training Kit，它描述了这 3 种服务的区别。

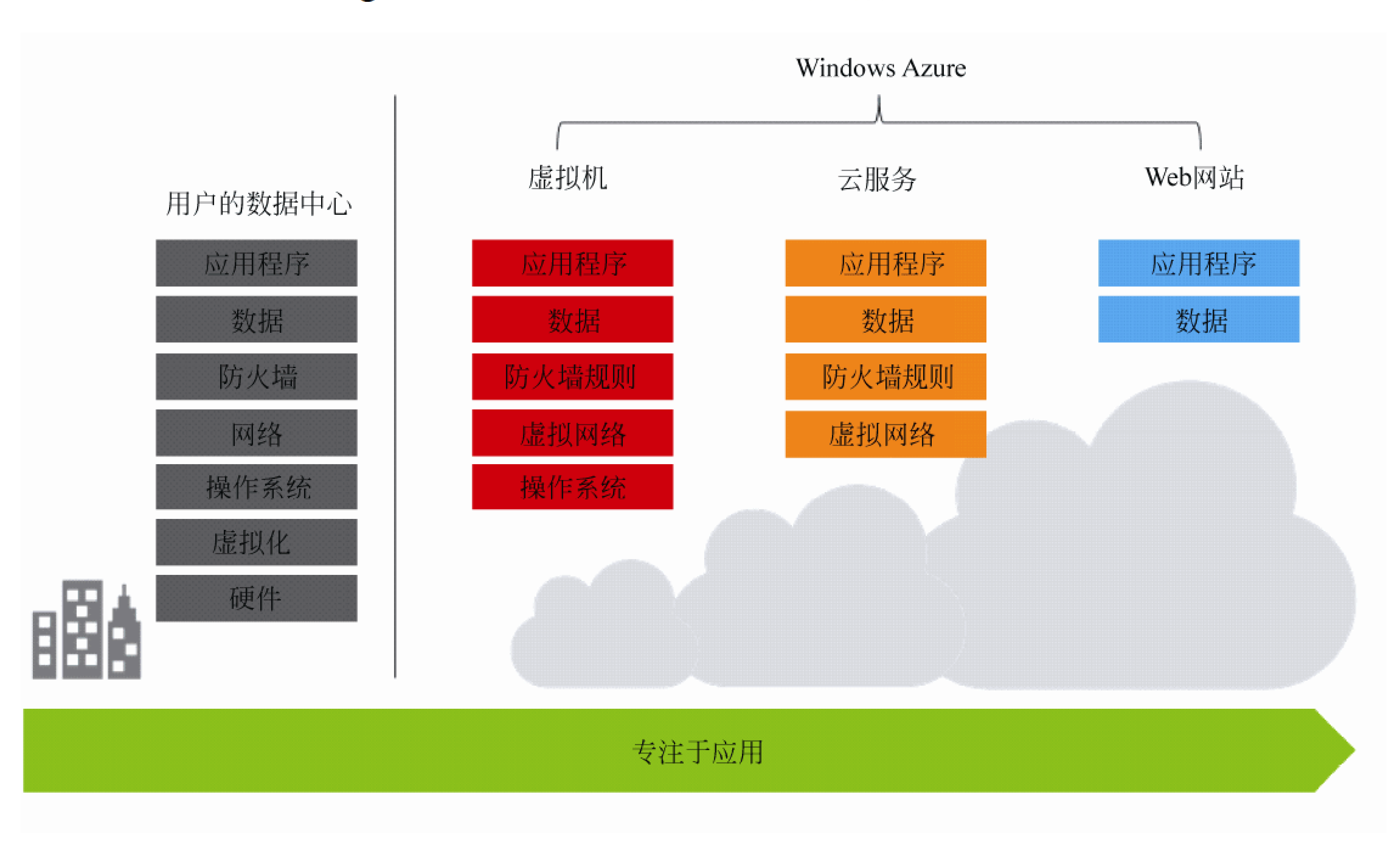


图 1-18　Azure 虚拟机、云服务、网站功能对比

3 种服务中，Azure 网站是最简单易用的选项，尤其适用于传统的 Web+数据库应用。当然，Microsoft Azure 云服务或 Microsoft Azure 的虚拟机也是运行用户的 Web 应用程序的不错的选择，尤其是在用户的 Web 应用需要一些额外的控制权限和定制服务时。但是，这种增加的控制是有代价的，它增加了应用程序的开发、管理和部署的复杂性。图 1-19 来源于 Microsoft Azure 的文档，描述了 3 种服务的易用性和可控制性的区别。

在许多情况下，Microsoft Azure 网站是最好的选择。它提供了简单灵活的开发、部署和管理选项，能够运行大并发量的 Web 网站，可以从 Web 应用库中快速创建一个新的网站、可以将现有的网站部署到 Microsoft Azure 网站，还可以使用 Microsoft Azure WebJobs 添加后台作业等等。

当需要下面的某一项功能时，考虑 Azure 云服务。

- 通过远程桌面连接到 Azure 的虚拟机。
- 使用 80 和 443 之外的网络端口访问用户的应用。

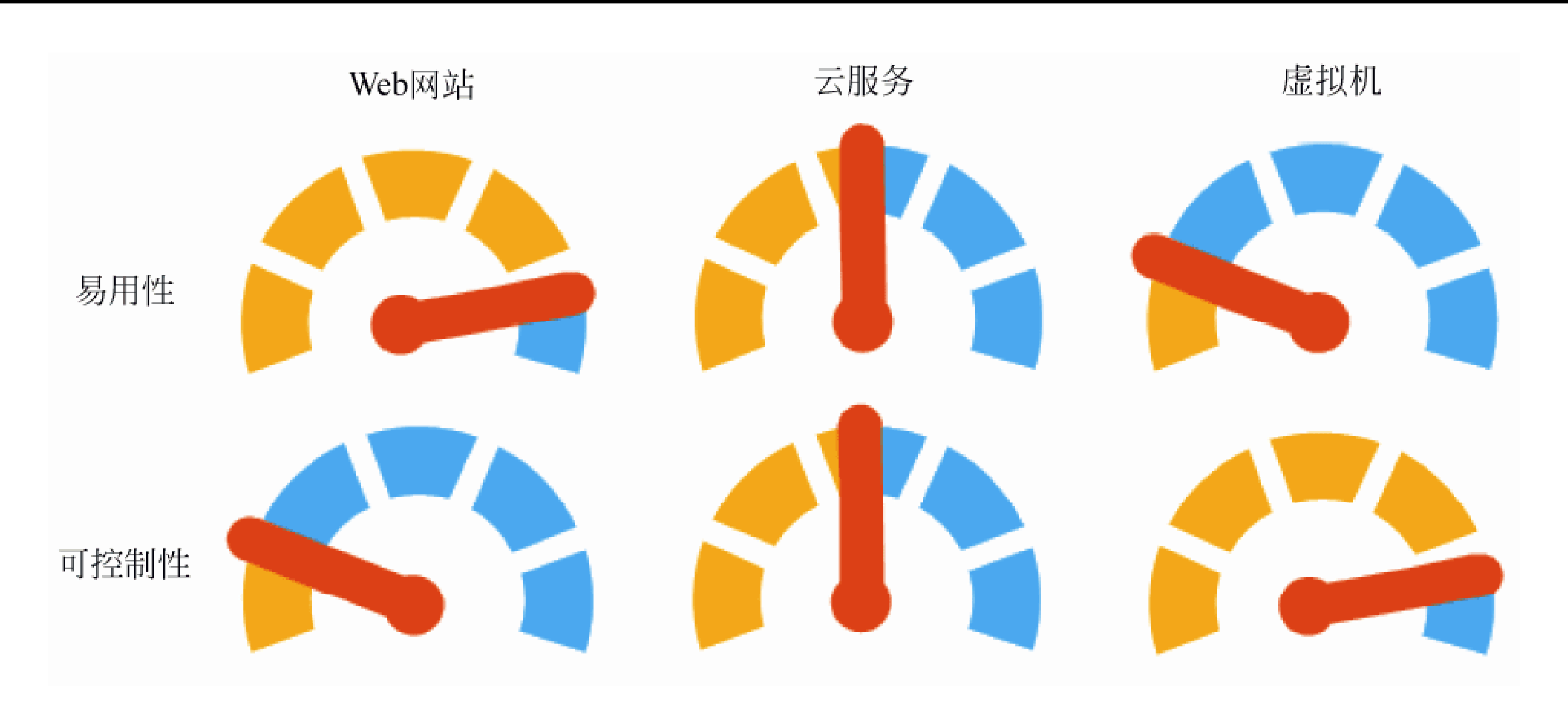


图 1-19 Azure 网站、云服务、虚拟机易用性与可控制性对比

- 完全控制 IIS，比如安装额外的 IIS 模块。
- 需要超过 10 台虚拟机实例运行用户的网站。
- 需要超小或者超大规格的虚拟机。
- 需要配置自定义防火墙规则。
- 将 Azure Blob 存储映射为本地驱动器。
- 用户的应用使用特殊 API 或者要求特殊的权限。
- 用户的应用依赖于 COM+/DCOM/COM 等。
- 用户的应用由多个逻辑层组成，而不是由 Web+数据库两层组成。

如果 Azure 云服务还不能满足用户的要求，比如，用户需要特殊版本的操作系统，或者需要 Linux 操作系统，那么用户可以选择 Azure 虚拟机。

1.5 参考文献与扩展阅读

1. Windows Azure Web Sites: An Architecture and Technical Deep Dive

Calvin Keaton 在 TechEd 2013 关于 Azure 网站的演讲视频。在演讲中揭示了 Azure 网站的设计和架构，并介绍了为什么 Azure 网站是托管 Web 应用的理想选择。

http://channel9.msdn.com/Events/TechEd/NorthAmerica/2013/WAD-B329/#fbid=

2. Scott Guthrie introduces Azure Web Sites and sets up Endpoint Monitoring

微软公司负责云计算业务的副总裁 Scott Guthire 详细介绍了 Azure 网站的负载均衡以及如何实时监视网站运行状况。

http://www.windowsazure.com/en-us/documentation/videos/websites-and-endpoint-monitoring-scottgu/

3. Azure Websites, Cloud Services and Virtual Machines Comparison

Azure 平台提供了多种托管 Web 应用的方案：Azure 网站、云服务和虚拟机。该文章

详细介绍了如何选择合适的托管方案。

http://www.windowsazure.com/en-us/documentation/articles/choose-web-site-cloud-service-vm/

4. What Is Azure?

该文档简单介绍了 Azure 提供的服务和相关的开发入门文档。

http://msdn.microsoft.com/zh-cn/library/windowsazure/dd163896.aspx

5. 私有云简介

私有云是为一个客户单独使用而构建的，提供了对数据、安全性和服务质量的最有效控制。构建私有云的公司拥有基础设施，并可以控制在此基础设施上部署应用程序的方式。私有云可部署在企业数据中心的防火墙内，也可以部署在一个安全的主机托管场所。该文章详细介绍了私有云的概念和特征。

http://baike.baidu.com/link?url=Fd_GsIiONnKaq3IJd3NOr62nuqWl2mhLgtTq5C0JM-IJNOGJ0t1omlNDYKw-wXk_fuksb6YtKCIV9dSHGXIiYa

6. 大企业私有云

大企业私有云利用云计算、企业计算、实时商业分析、大数据管理等多项新技术，探索创新业务发展模式，整合社会化存量资源，提高资源利用率，降低资源消耗，支撑绿色和可持续发展，满足企业管理变革快、服务质量高、投入成本低的经营诉求，全面面向客户服务和价值创造，打造健康、多赢、可持续的商业生态环境。该文章介绍了大企业私有云的概念、架构和优势。

http://baike.baidu.com/link?url=oF021KuV_HRkpCxXw6E42otnNHXvdZu02C8xsKBfsxNLbvDeqHtcYxn-I1yQb1V_zhbngDcah0BtylJsgx4wNK

7. 云计算

云计算是基于互联网的相关服务的增加、使用和交付模式，通常涉及通过互联网来提供动态易扩展且经常是虚拟化的资源。该文档介绍了云计算的定义、特点以及服务形式相关的基本概念等。

http://baike.baidu.com/link?url=zF4KhJaZ1nVABZqvXTZ90sN8XV581jR-IZzPXySrnlmlzHVgC4ym00HzshovgaFMJRNN64REYaJtH2aqwuDXcK

8. 认识云计算

我们正在经历着一个前所未有的变革时代。信息技术的不断创新也推动着各行业的业务创新。快速成长与转型的现代企业需要一个动态 IT 基础结构来支撑，云计算的出现则是构建动态 IT 基础结构的最有效方法。

云计算的最终目标是将计算、服务和应用作为一种公共设施提供给公众，使人们能够像使用水、电、煤气和电话那样使用计算机资源。微软公司认为，未来的互联网将会是一个由“云+端”组成的世界。该文章详细介绍了云计算的基本概念和术语。

http://wacnstorage.blob.core.chinacloudapi.cn/marketing-resource/documents/1%20%E8%AE%A4%E8%AF%86%E4%BA%91%E8%AE%A1%E7%AE%97.pdf

9. Cloud Computing

维基百科关于云计算的介绍。该文档介绍了云计算的定义、特点以及服务形式的基本概念等。

http://en.wikipedia.org/wiki/Cloud_computing

10. Introduction to Microsoft Azure: the cloud operating system（Mark Russinovich, Build 2011）

Mark，微软公司云计算平台 CTO 关于 Windows Azure 平台的介绍。

http://channel9.msdn.com/Events/Build/BUILD2011/SAC-852F

11. Introducing Microsoft Azure（David Chappell,October 2012）

Azure 白皮书，详细介绍了 Azure 的设计、概念和服务。

http://go.microsoft.com/?linkid=9682907&clcid=0x409

12. Windows Azure Poster

Azure 海报，涵盖了 Azure 提供的各项服务。

http://www.microsoft.com/ZH-CN/download/details.aspx?id=35473

13. Azure Websites

Azure 官方网站，包含产品介绍、开发、维护等文档。

http://www.windowsazure.com/en-us/services/web-sites/

14. IIS ARR

IIS ARR 是一个工作在 HTTP 层的负载均衡模块。它是 Microsoft Azure 前端服务器的最重要的组件。建议读者阅读下面的文章，以更好地理解前端服务器的工作原理。

http://www.iis.net/downloads/microsoft/application-request-routing

15. Microsoft Web Farm Framework

Microsoft Web Farm Framework 提供了部署、扩展和管理多台 Web 服务器的解决方案。Azure 网站的架构借鉴了该方案。可以参考下面的文章以更好的了解 Azure 网站的架构。

http://www.iis.net/learn/web-hosting/microsoft-web-farm-framework-20-for-iis-7

16. Microsoft Azure Internals（Mark Russinovich, Build 2012）

Mark Russinovich 深入介绍了 Azure 平台的架构以及基本概念。

http://channel9.msdn.com/Events/Build/2012/3-058

第 2 章　管理、配置和监视 Azure 网站

本章详细介绍确保 Azure 网站平稳运行的管理任务，包括创建、管理、配置、监视网站。多数管理任务都可以通过 Azure 管理门户网站完成，非常简单、直观。本章用了很大篇幅详细介绍 Azure 网站配置中的难点：配置自有域名和 SSL 证书。除此之外，本章同时详细介绍 Azure 网站的 DNS 和 SSL 功能的内部实现，以帮助用户更好地了解如何配置自有域名和 SSL 证书。

2.1　Microsoft Azure 管理门户网站

Microsoft Azure 管理门户网站是一个简洁的 HTML 5 网站，支持所有主要的浏览器和设备（IE9 及以上，Firefox，Chrome 和 Safari 的最新版本）。

通过 Microsoft Azure 管理门户网站，开发人员和 IT 专业人员可以部署、配置、监控和管理他们的 Microsoft Azure 应用程序；深入了解部署在 Azure 中的资源状态和使用情况。Azure 管理门户网站提供了丰富的监控选项。管理门户网站支持所有 Microsoft Azure 服务。

Microsoft Azure 全球管理门户网站：

https://manage.windowsazure.com

Microsoft Azure 中国区管理网站（由世纪互联运营）：

Https://manage.windowsazure.cn

图 2-1 是全球管理门户网站的截屏，主要由 4 部分组成：



图 2-1　Azure 管理门户

1. 导航面板

左侧的导航面板列出了所有可以通过门户网站进行管理的服务，包括网站、虚拟机、移动服务、云服务、SQL 数据库、存储、HDInsight、SQL 报表、备份/恢复、服务总线、媒体服务、BizTalk 服务、Active Directory、缓存服务、虚拟网络和流量管理器等。（目前 Microsoft Azure 中国区网站只支持部分核心服务。）

2. 订阅信息

顶部的订阅栏显示当前订阅信息，通过订阅栏可以查看账单、联系微软公司技术支持部门、更改密码和快速导航到其他 Microsoft Azure 网站（价格、文档和论坛等）。

3. 命令栏

底部的命令栏根据选择的服务提供对应的命令按钮，比如新建、启动、停止和删除资源，保存/取消配置修改。

4. 主面板

中央的主面板显示当前服务的具体信息，通常包括仪表盘、资源监视和配置选项等。

2.1.1 创建 Azure 网站

通过 Microsoft Azure 管理门户网站，只需点击几次鼠标即可轻松创建一个网站。Microsoft Azure 网站提供了 3 个创建网站的选项。

1. 快速创建

该选项允许用户快速创建一个空的网站。网站创建后，Microsoft Azure 自动为用户的网站添加一个默认文件 hostingstart.html。

2. 自定义创建

自定义创建选项允许用户创建网站的同时创建一个新的数据库或者关联一个已有的数据库。数据库可以是 SQL 数据库或 MySQL 数据库。如果用户需要移植一个包含后台数据库的网站应用，可以选择该选项。同时，该选项还允许用户指定源代码控制选项，比如 GitHub 或 Team Foundation Server （TFS）。如果用户当前正在使用版本管理软件管理用户的代码，该选项提供了一个快速的方法来设置用户的 Microsoft Azure 网站以进行部署。Azure 网站目前支持从如下系统中部署网站应用：

- Visual Studio Online。
- 本地 Git。
- GitHub。
- Dropbox。
- Bitbucket。

- CodePlex。
- 外部存储库（支持 Git/Bitbucket）。

3. 从库中创建

Microsoft Azure 网站包括一个丰富的 Web 应用库。该库中包含了很多主流网站框架，比如 Drupal、WordPress 等。如果用户需要一个 WordPress 网站，可以从库中选择 WordPress，这样就可以快速地在 Microsoft Azure 网站中建立一个自己的基于 WordPress 的网站。

如果有兴趣将自己的 Web 应用提交到 Web 应用库中，请参考下面的文档：

http://www.microsoft.com/web/gallery/

2.1.2　创建网站的流程

在后台，创建一个 Azure 网站的流程如下：

（1）检查网站名称是否可用。

（2）检查订阅网站数量是否已经达到上限。如前所述，在每个数据中心，每个订阅可以有 10 个免费模式网站、100 个共享模式网站和无限的基本模式和标准模式网站。

（3）通过 Azure DNS 服务注册网站 DNS（*sitename*.azurewebsites.net）。

（4）在运行时数据库生成网站相关记录。

（5）创建网站文件存储目录，并将默认文件（hostingstart.html）复制到该目录。

（6）如果是自定义创建，Azure 网站会自动创建并关联指定的数据库和配置版本控制选项，并将之与网站相关联。

（7）如果是从库中创建，Microsoft Azure 通过 Web Deploy 将选择的应用部署到网站。

2.1.2.1　快速创建第一个网站

在 Microsoft Azure 中，只需要 1 分钟即可拥有一个站点。快速创建网站的步骤如下：

（1）登录到 Microsoft Azure 管理门户网站。

（2）单击左下角的“新建”按钮，选择“计算”→“网站”。

（3）选择“快速创建”，如图 2-2 所示，输入如下信息：



图 2-2　快速创建一个网站

① URL：输入要创建的网站名称。该名称必须是未被注册的名称。

② WEB 宿主计划：选择“创建新的 Web 宿主计划”。

③ 区域：选择一个数据中心。选择原则是尽量靠近自己的客户。

（4）单击右下方的“创建网站”按钮。

（5）稍候片刻，网站即可创建成功。

（6）此时，即可通过 MyQuickCreateSite.azurewebsites.net 来访问您刚刚创建的网站。

2.1.2.2 自定义创建网站

自定义方式创建一个站点允许您在创建网站的同时创建一个 MySQL 或者 MS SQL 数据库，并将其与创建的网站关联。下面的步骤创建一个网站，并创建一个 MySQL 数据库。

（1）登录到 Microsoft Azure 管理门户网站。

（2）单击左下角的“新建”按钮，选择“计算”→“网站”。

（3）选择“自定义创建”，如图 2-3 所示，输入如下信息：



图 2-3 自定义创建一个网站

① URL：输入要创建的网站名称。该名称必须是未被注册的名称。

② WEB 宿主计划：可以选择“已有的 Web 宿主计划”，或者选择“创建新的 Web 宿主计划”。在这里选择“创建新的 Web 宿主计划”。

③ 区域：选择一个数据中心。选择原则是尽量靠近自己的客户。

④ 数据库：有以下选项，在这里选择创建新的 MySQL 数据库：

- 使用现有 SQL 数据库。
- 创建新的 SQL 数据库。

- 使用现有的 MySQL 数据库。
- 创建新的 MySQL 数据库。

⑤ 数据库连接字符串名称：这里输入 DefaultConnection。

（4）如图 2-4 所示，指定 MySQL 数据库的名称和区域，单击“完成”。



图 2-4　创建 MySQL 数据库

2.1.2.3　从库中创建 DNN 站点

Microsoft Azure 网站支持一个丰富的 Web 应用库，包括 DNN、Orchard、WordPress 和 nopCommerce 等热门 Web 应用。只需几分钟，即可轻松从 Web 应用库中创建一个网站。下面是从库中创建一个 DNN 网站的步骤。

（1）登录到 Microsoft Azure 管理门户网站。

（2）单击左下角的“新建”按钮，选择“计算”→“网站”。

（3）选择“从库中”，如图 2-5 所示，在列表中选择 DNN Platform，单击“下一步”。



图 2-5　从 Web 应用库中创建网站

（4）在“站点设置”页面，如图 2-6 所示，提供如下信息，单击“下一步”。

① URL：提供要创建的网站名称，必须是未被注册的网站。

② 数据库：可以选择“使用现有 SQL 数据库”，或者“创建新的 SQL 数据库”，在这里选择“创建新的 SQL 数据库”。

③ WEBSACLEGROUP：可以选择“已有的 Web 宿主计划”，或者选择“创建新的 Web 宿主计划”。在这里选择“创建新的 Web 宿主计划”。

④ 区域：选择一个数据中心。选择原则是尽量靠近自己的客户。

（5）指定数据库设置。如图 2-7 所示，指定数据库名称、数据库服务器名称、数据库服务器的登录名和登录密码。可以选择“使用已有的数据库服务器”或者“新建 SQL 数据库服务器”，在这里选择“新建 SQL 数据库服务器”。



图 2-6 DNN 网站设置　　图 2-7 指定数据库设置

（6）单击“完成”按钮，稍候片刻，DNN 网站以及相应的数据库即可创建成功。

（7）此时即可访问刚刚建立的 DNN 网站。如图 2-8 所示，进入 DNN 网站的安装页面。

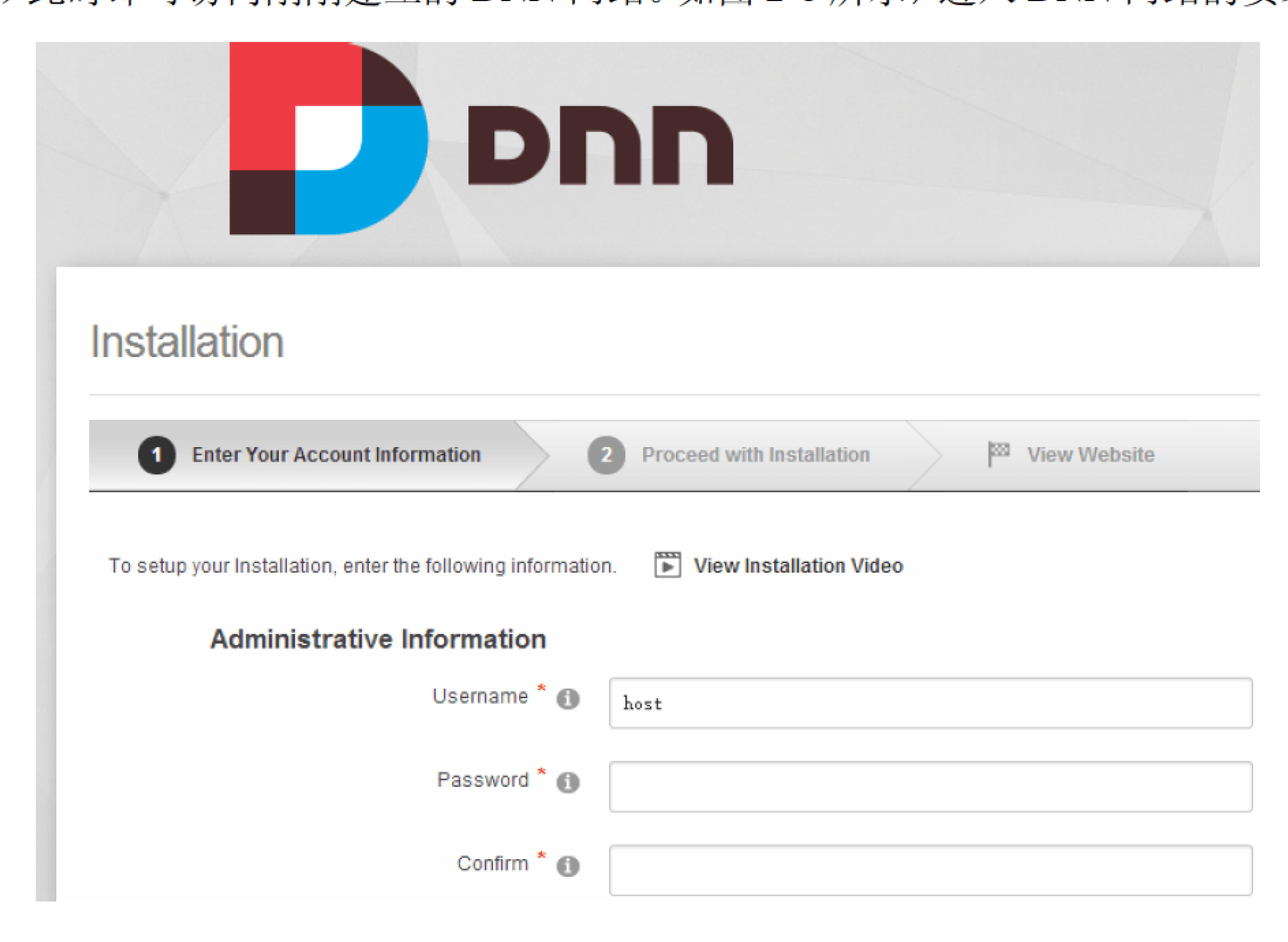


图 2-8 DNN 安装页面

2.2　管 理 网 站

Azure 管理门户网站提供了一组页面用于管理用户的网站。

1．快速开始页面

快速开始页面包含以下部分：

1）获取工具

该部分提供了 WebMatrix 和 Microsoft Azure SDK 的下载链接。

2）发布应用程序

该部分包含下载发布配置文件、重置部署凭据、添加新的部署槽和有关过渡发布的信息。

3）集成源代码管理

该部分用于设置和管理从源代码控制工具部署网站，支持 TFS、CodePlex、GitHub、Dropbox、bitbucket 或本地的 Git。

图 2-9 描述了快速开始页面的基本功能。

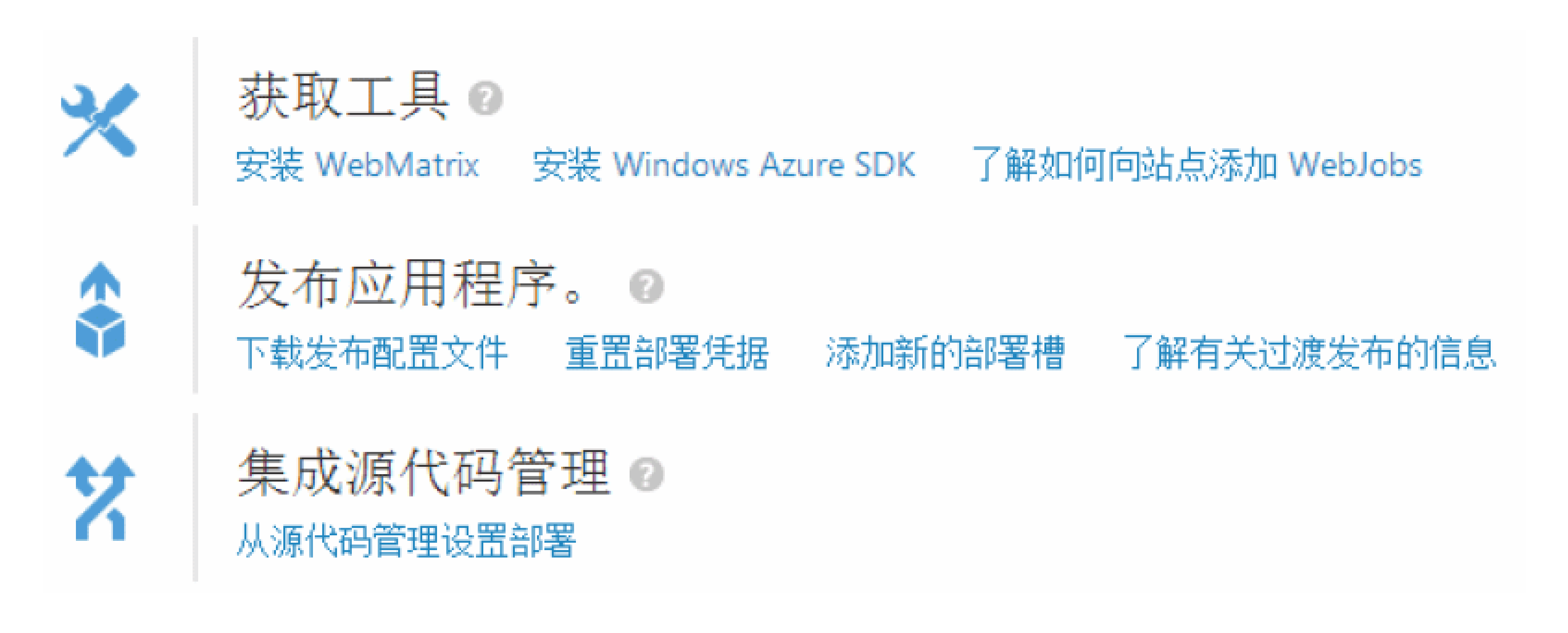


图 2-9　网站快速开始页面

2．仪表板

仪表板包含网站的基本信息，包括以下内容：

1）性能图表

性能图表包含如下性能指标：

- CPU 时间。
- HTTP 服务器错误。
- 请求。
- 输入数据量。
- 输出数据量。

2）Web 端点状态

在该页面可以配置 Web 端点的当前状态。在本章会讨论如何配置 Web 端点监视。

3）自动缩放状态

Azure 网站可以根据网站的资源使用率自动缩放。在本章会讨论如何配置自动缩放。

4）使用概览

给出网站的资源使用情况，包括：

- CPU 时间。
- 输出数据量。
- 文件系统存储空间。
- 内存用量。

5）链接的资源

该页面显示网站链接的资源，可以是 MySQL 数据库、SQL Azure 和 Azure 存储。

6）速览

速览页面包括如下信息：

- 状态。
- 操作日志。
- 虚拟 IP 地址。
- 网站 URL。
- 计算模式。
- FTP 主机名。
- 部署用户。
- FTP 日志。
- 数据中心。
- 订阅名称。
- 订阅 ID。

3. 部署

部署页面只有在配置了“从源代码管理库部署”才会可见。部署页面包含所有已经部署的版本的信息。

4. 监视器

监视器页面提供了一个显示网站当前资源使用量的性能图表。默认情况下，它与仪表板的性能图表显示的信息相同。但是，可以在监视器页面添加更多性能指标进行监视。

5. Web 作业

Web 作业页面用于管理 Web 作业。利用 Web 作业，可以创建运行在后台的程序，用于处理比较耗时的任务。关于 Web 作业的更多信息，请参考下面的文档：

How to Use the WebJobs Feature in Microsoft Azure Web Sites

http://www.windowsazure.com/en-us/documentation/articles/web-sites-create-web-jobs/

6．配置

配置页面用于修改、设置网站和应用相关的配置。随后会具体讨论如何修改网站/应用配置。

7．缩放

在缩放页面，可以配置网站宿主计划模式：免费、共享、基本和标准。同时，可以配置对应的网站实例数目和自动缩放功能。在本章会详细讨论网站缩放。

8．链接的资源

可以在该页面管理链接的资源，比如 MySQL、SQL Azure 和 Azure 存储。

9．备份

在备份页面，可以备份和还原网站。

2.3　网 站 配 置

可以通过 Microsoft Azure 管理门户网站修改 Azure 网站的配置。Azure 网站的配置选项分为如下几类。

1．常规

（1）.NET Framework 版本。设定 Web 应用程序需要的.NET Framework 的版本。目前支持.NET 3.5 和.Net 4.5。

（2）PHP 版本。设置 Web 应用程序需要的 PHP 版本，目前支持 5.3、5.4 和 5.5 版。如果不需要，可以关闭 PHP 功能。

（3）托管管道模式。IIS 工作进程模式，有两个选项：经典模式和集成模式。默认是集成模式。与运行在本地的 IIS 相同，只有当用户的应用不兼容集成模式时，才应该选择经典模式。

（4）平台。指定 IIS 工作进程的运行平台（32 位或 64 位）。免费和共享模式只支持 32 位 IIS 工作进程。在标准模式下，支持 32 位和 64 位。

（5）Web Socket。打开和关闭 Web Socket 功能，默认为关闭。

（6）始终可用。对于运行在本地的 IIS 系统，默认情况下，如果在 20 分钟内没有任何请求，IIS 将自动关闭对应的工作进程以节省系统资源。对于 Azure 网站而言，如果在 20 分钟内没有任何请求，Azure 网站将被从工作机器中移除以节约系统资源。如果再有新的请求到达时，网站会再次被创建在工作机器上。通常，这会导致短暂的性能下降。在标准模式下，可以启用始终可用功能，以保证网站不会因为空闲而被移除。启用始终可用功能后，系统自动定期访问网站根目录，确保网站不会因为闲置而被关闭。另外，如果配置了连续运行的 Web Job，应该启用始终可用功能以保证 Web Job 连续可靠地运行。

（7）在 Visual Studio Online 中在线编辑代码。选择开启 Visual Studio 实时代码在线编辑功能。启用该功能后，仪表板选项卡的快速概览部分将显示在 Visual Studio Online 中编辑链接。单击该链接将打开 Visual Studio Online 在线编辑器，可以在线编辑代码。

注意：如果启用了“从源代码部署管理”，它可能会覆盖在 Visual Studio 中修改的代码。因此，如果想直接在 Visual Studio 在线编辑网站内容，最好不要使用“从源代码部署管理”。

2. 安全证书

在标准模式下，可以上传自有域名的 SSL 证书。已经上传的证书会在这里列出。支持通配符（证书主题有星号，比如*.contoso.com）。通配符证书只需要上传一次，便可应用于任何绑定到*.contoso.com 的网站，比如 it.contoso.com、hr.contoso.com 等所有符合要求的网站。只有当证书没有被任何网站绑定使用时，证书才可以被删除。

3. 自有域名

为 Azure 网站绑定自有域名。共享、基本和标准模式支持自有域名，免费模式不支持该功能。

4. SSL 绑定

Azure 网站支持两种方式的 SSL 绑定模式：SNI SSL 和基于 IP 的 SSL。

5. 应用程序诊断

如果用户的应用使用了 System.Diagnostics.Trace 来记录应用诊断信息，那么可以在这里设置应用诊断跟踪选项：

（1）应用程序日志记录（文件系统）。将应用诊断日志写入网站的文件系统中。当启用时，文件系统日志记录持续期为 12 小时。诊断日志文件可以通过 FTP 下载。FTP 的具体信息显示在仪表盘选项卡上。

注意：如果日志文件过大，可能会导致超出文件系统存储配额。

（2）应用程序日志记录（表存储）。将应用程序日志写入 Microsoft Azure 表存储账户。记录到表存储的日志没有时间和大小限制。

（3）应用程序日志记录（Blob 存储）。将应用程序日志写入 Microsoft Azure Blob 存储账户。记录到 Blob 存储的日志没有时间和大小限制。

6. 网站诊断

设置选项用于收集您的网站的诊断信息，其中包括以下选项：

（1）Web 服务器日志。指定是否启用网站的 Web 服务器日志记录。 Web 服务器的日志都保存为 W3C 扩展日志文件格式。可以将日志保存到 Microsoft Azure 存储或文件系统。

如果选择保存到文件系统，可以设置日志文件最大可用磁盘空间。最小为 25MB，最

大为 100MB。默认值是 35MB。当达到限额时，系统自动覆盖最早的日志文件。如果需要保留更多日志，可将日志保存到 Microsoft Azure 的 Blob 存储。

默认情况下，Microsoft Azure Blob 存储中的 Web 服务器日志不会被删除。要指定在一段时间之后日志将被自动删除，选择“设置保留”并输入日志保留天数。

（2）详细的错误消息。指定是否记录网站错误的详细信息。如果启用，详细的错误消息以 HTML 格式保存到文件系统。可以通过 FTP 导航到/LogFiles/ DetailedErrors 目录下载。该功能与 ASP.NET 的 Yellow Page 错误信息类似。

（3）失败请求跟踪。指定是否启用 IIS 失败请求跟踪。如果启用，失败请求跟踪输出写入到 XML 文件，并保存到文件系统。可以通过 FTP 导航到/LogFiles/W3SVC siteId 来下载。在该文件夹下有一个 freb.xsl 文件，请同时下载该文件。在浏览器中查看失败请求跟踪文件时，XSL 文件提供格式化和过滤功能。

7. 远程调试

将此选项设置为开，可以通过 Visual Studio 2012 或 Visual Studio 2013 的远程调试器功能直接连接到用户自己的网站进行远程调试。

注意：远程调试在启用 48 小时后自动关闭，网站名称或用户名称不能超过 20 个字符。

8. 监控

仅标准模式的网站可用。该功能允许从多达 3 个地理分布位置测试 HTTP 或 HTTPS 端点的可用性。对于标准模式的网站，下列情形表示监视测试失败：

（1）HTTP 响应代码大于或等于 400。

（2）页面响应时间超过 30s。

警报功能可以在监视失败时自动发送邮件给管理员。

9. 开发人员分析

支持第三方分析软件，如使用 New Relic 管理分析应用程序性能。

10. 应用程序设置

指定应用程序设置名称及对应的值，在启动时 Web 应用程序会自动加载这些设置。对于.NET 网站，这些设置会在运行时被注入到 web.config 的 AppSettings 配置节点。如果 web.config 已经定义了同名配置，则覆盖已有的设置。对于 PHP 和 Node.js，这些名称/值对将被设置为环境变量，供应用程序在运行时访问。

11. 连接字符串

设置和查看连接字符串。用于.NET 的网站，连接字符串会在运行时被注入到 web.config 的 connectionString 配置节点。如果 web.config 已经定义了同名配置，则覆盖已有的设置。对于 PHP 和 Node.js，这些名称/值对将被设置为环境变量，可以在运行时访问。环境变量的前缀如下：

- SQL Server：SQLCONNSTR_。
- MySQL：MYSQLCONNSTR_。
- SQL Azure 数据库：SQLAZURECONNSTR_。
- 自定义：CUSTOMCONNSTR_。

例如，如果一个 MySQL 连接字符串命名为 connectionstring1，则需要通过环境变量 MYSQLCONNSTR_connectionString1 来访问。

注意：当将数据库资源链接到一个网站时会生成连接字符串。在配置管理页面上只能查看而不能修改该连接字符串。

12. 默认文档

设置网站的默认文档列表。如果默认文档设置为 default.htm，当用户访问 http://www.yoursite.com 时，用户将被指向 http://www.contoso.com/default.htm。可以设置多个默认文档，排在最前面的优先级最高。

13. 处理程序映射

添加自定义脚本处理器来处理特定文件扩展名的请求。比如，用户有自己的 Fast CGI 来处理某些特殊的文档，则可以通过该功能配置自己的 Fast CGI。指定脚本处理器时必须使用绝对路径（路径 d:\home\site\wwwroot 是网站的根目录）。

14. 虚拟应用程序和目录

配置网站下的子应用程序和虚拟目录，指定每个虚拟目录及其相对于站点根目录的物理路径。

2.4 网站备份与恢复

Microsoft Azure 网站备份和恢复功能可让用户轻松地手动或自动创建网站的备份。通过备份可以将网站恢复到以前的状态，也可以使用备份创建一个新的网站。

Microsoft Azure 的网站备份和恢复功能仅支持标准模式的站点。使用备份和恢复功能，需要一个 Microsoft Azure 存储账户。网站备份将被存储在 Microsoft Azure 存储中。Microsoft Azure 网站备份包括以下信息：

（1）网站配置。

（2）网站文件内容。

（3）任何连接到网站的 SQL Server 或 MySQL 数据库（可选）。

2.4.1 手动备份网站

手动备份网站的步骤如下：

（1）登录到 Microsoft Azure 管理门户网站。

（2）在网站管理页面单击“备份”，如图 2-10 所示。



图 2-10　网站备份

（3）在“备份”页面，选择备份使用的存储账户，如图 2-11 所示。如果还没有存储账户，需要先创建一个存储账户，该存储账户用于保存备份的数据。

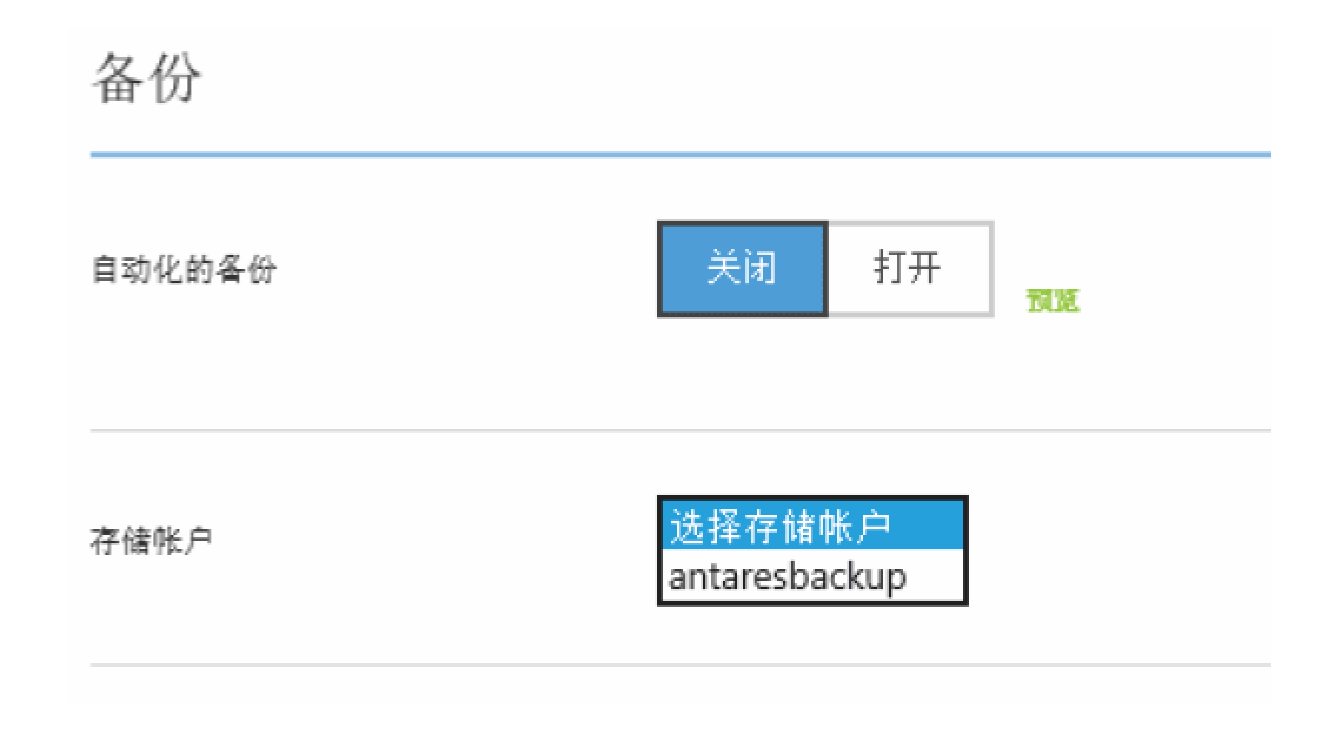


图 2-11　选择存储账户

（4）选择需要备份的数据库，如图 2-12 所示。



图 2-12　选择要备份的数据库

（5）在底部命令栏单击“立即备份”，如图 2-13 所示。

（6）在底部通知区，会看到网站备份已经成功启动的消息，如图 2-14 所示。



图 2-13　立即备份　　图 2-14　网站备份启动消息

备份是异步操作，启动后，在后台自动进行备份工作。目前，只允许每天两次手动备份。

2.4.2　自动备份网站

自动备份网站的步骤如下：

（1）在备份页面，打开“自动化的备份”选项，如图 2-15 所示。



图 2-15　自动备份

（2）指定备份使用的存储账户。如果还没有存储账户，需要先创建一个存储账户。

（3）指定备份频率和开始日期，图 2-16 的设置表示从 2014 年 3 月 9 日起，每 10 天备份一次。

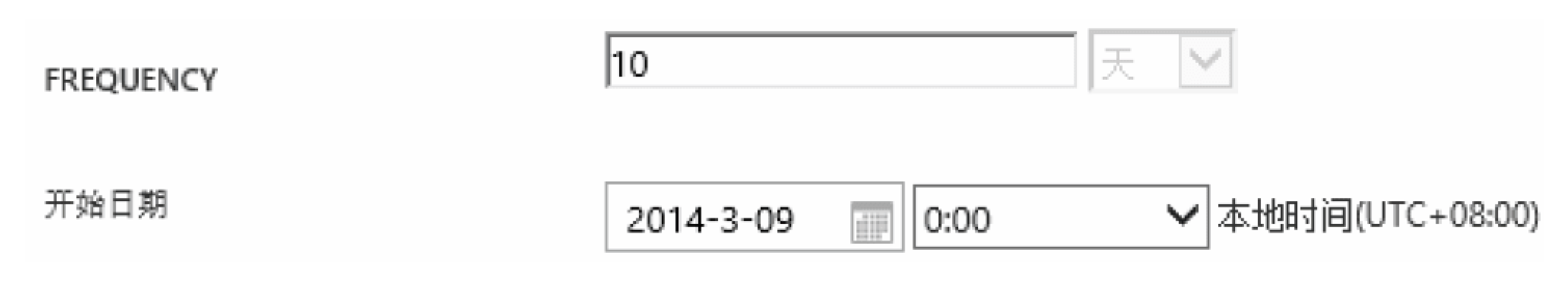


图 2-16　自动备份频率

（4）选择需要保存的数据库。

（5）单击命令栏的“保存”，保存设置，如图 2-17 所示。



图 2-17　保存自动备份设置

2.4.3　备份的管理

网站备份保存在存储账户中，存放在一个名为 websitebackups 的容器中。每个备份由一个包含备份数据的.zip 文件和包含内容清单的.xml 文件组成。.zip 和.xml 备份文件名由“网站名称_备份的时间戳”组成。时间戳包含日期的格式为 YYYYMMDD（不带空格的数字），再加上 UTC 格式的 24 小时（例如 drumboy_201403091325.zip）。如果备份文件被删除，将无法从备份中恢复网站。可以通过 Azure Explorer 或者任意其他 Azure 存储浏览器（比如 Visual Studio）来查看管理备份文件。图 2-18 展示了在 Cerebrata Azure Explorer 中的备份文件。可以访问 http://www.cerebrata.com 网站，下载免费的 Azure Explorer 软件。

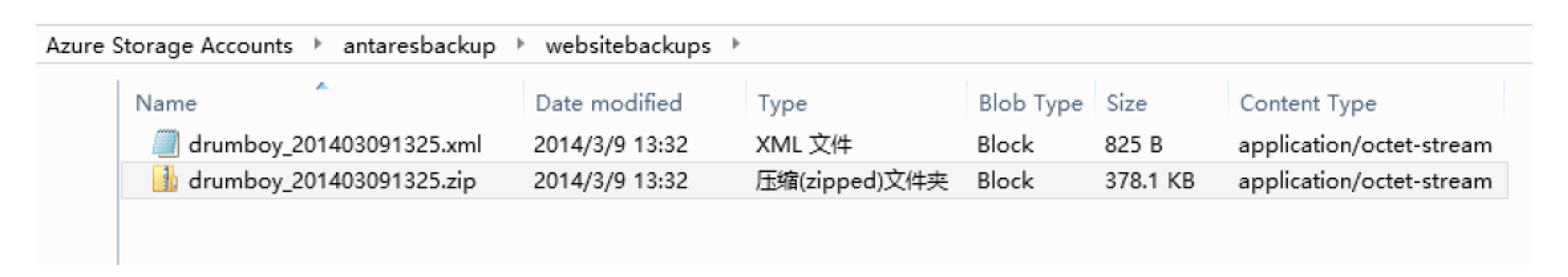


图 2-18　Azure Explorer

2.4.4　从备份中恢复网站

从备份中恢复网站的步骤如下：

（1）在备份选项卡，单击门户网站页面底部命令栏的“立即还原”。

（2）在“立即还原”对话框中有两个选项。

① 如图 2-19 所示，选择“该网站的上一个备份”作为备份源。默认情况下，会列出所有的备份。

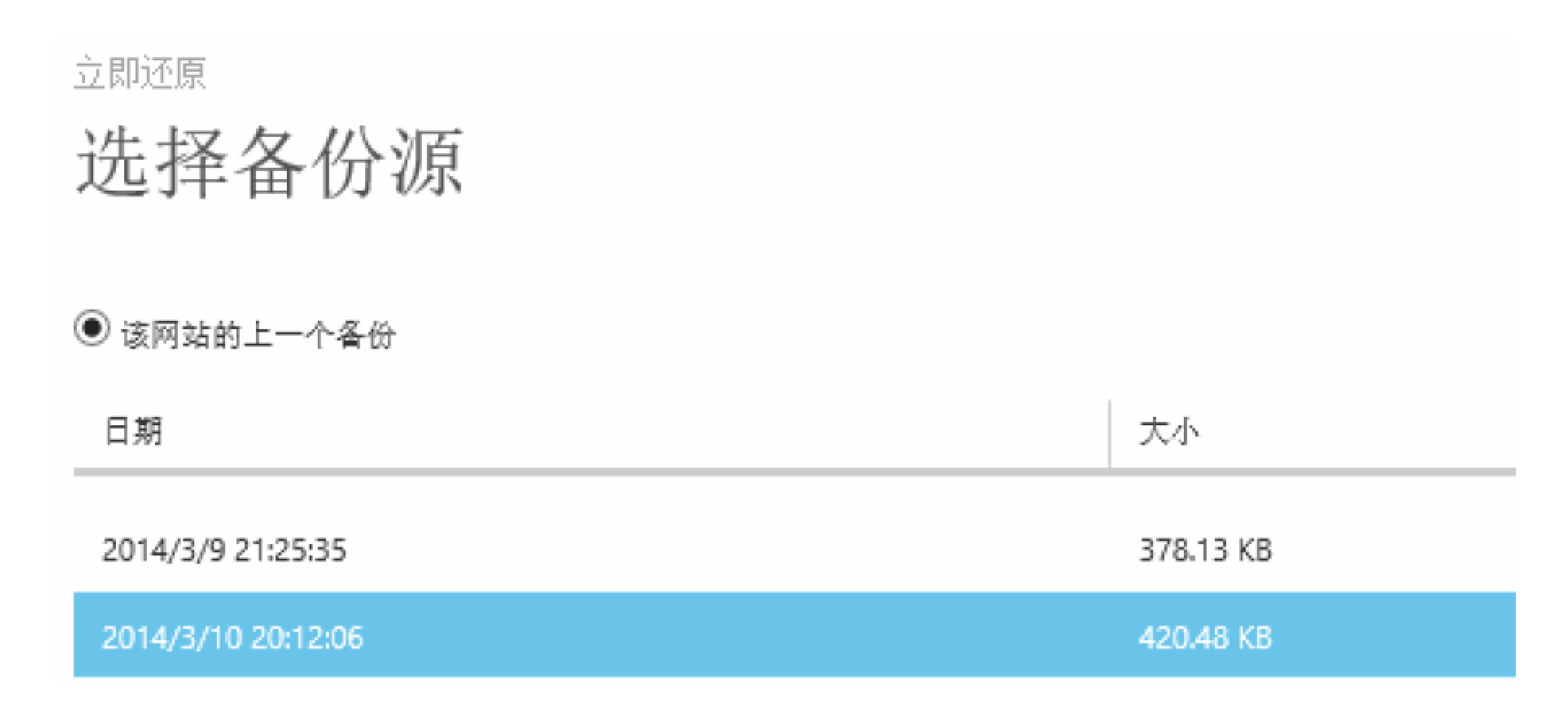


图 2-19　选择一个备份

② 如图 2-20 所示，可以选择存储账户文件。选择一个存储账户后，浏览并指定备份文件。

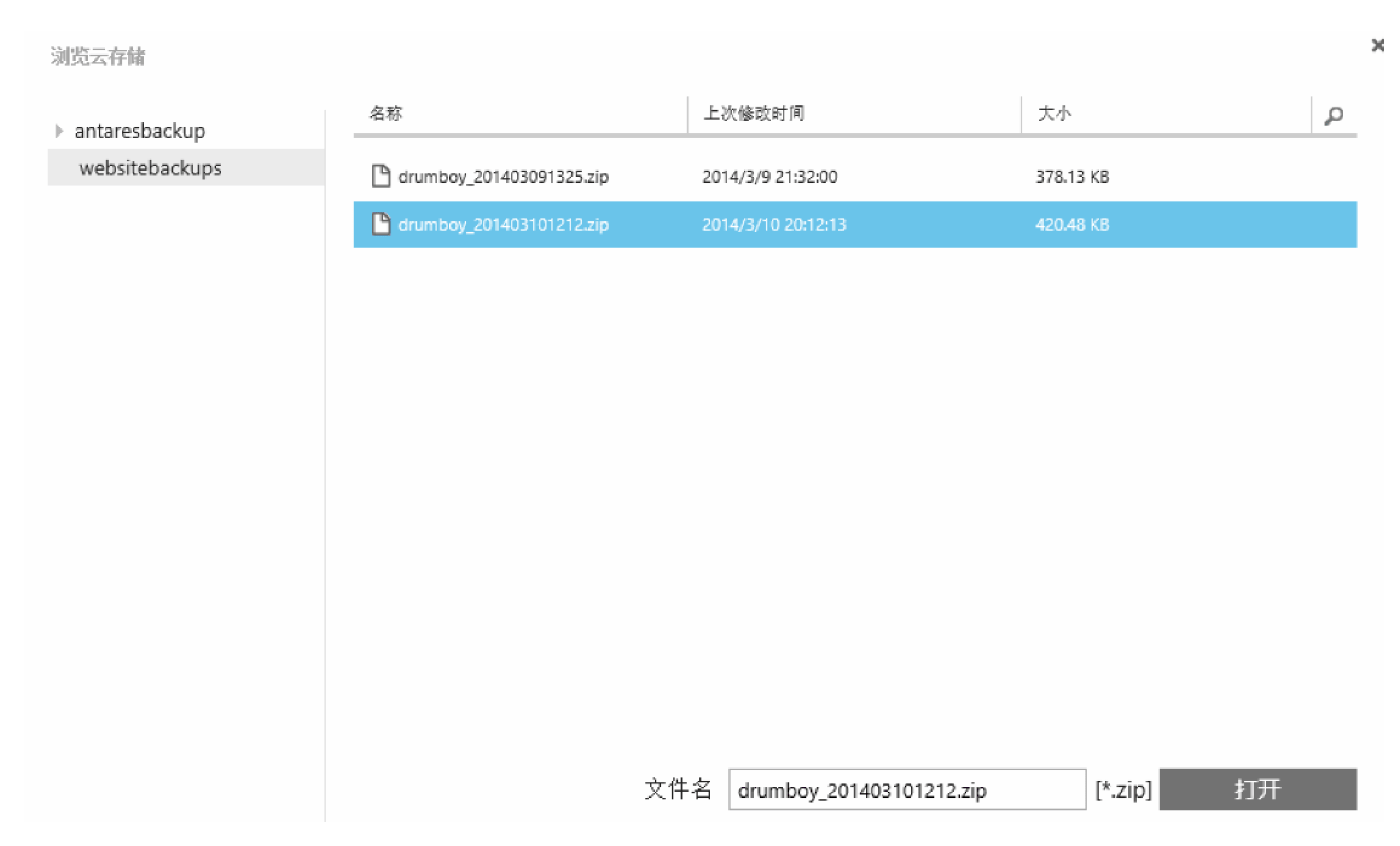


图 2-20　选择存储账户

（3）单击下一步箭头，设定网站还原设置。

如图 2-21 所示，可以选择还原到当期网站或者新建网站实例。

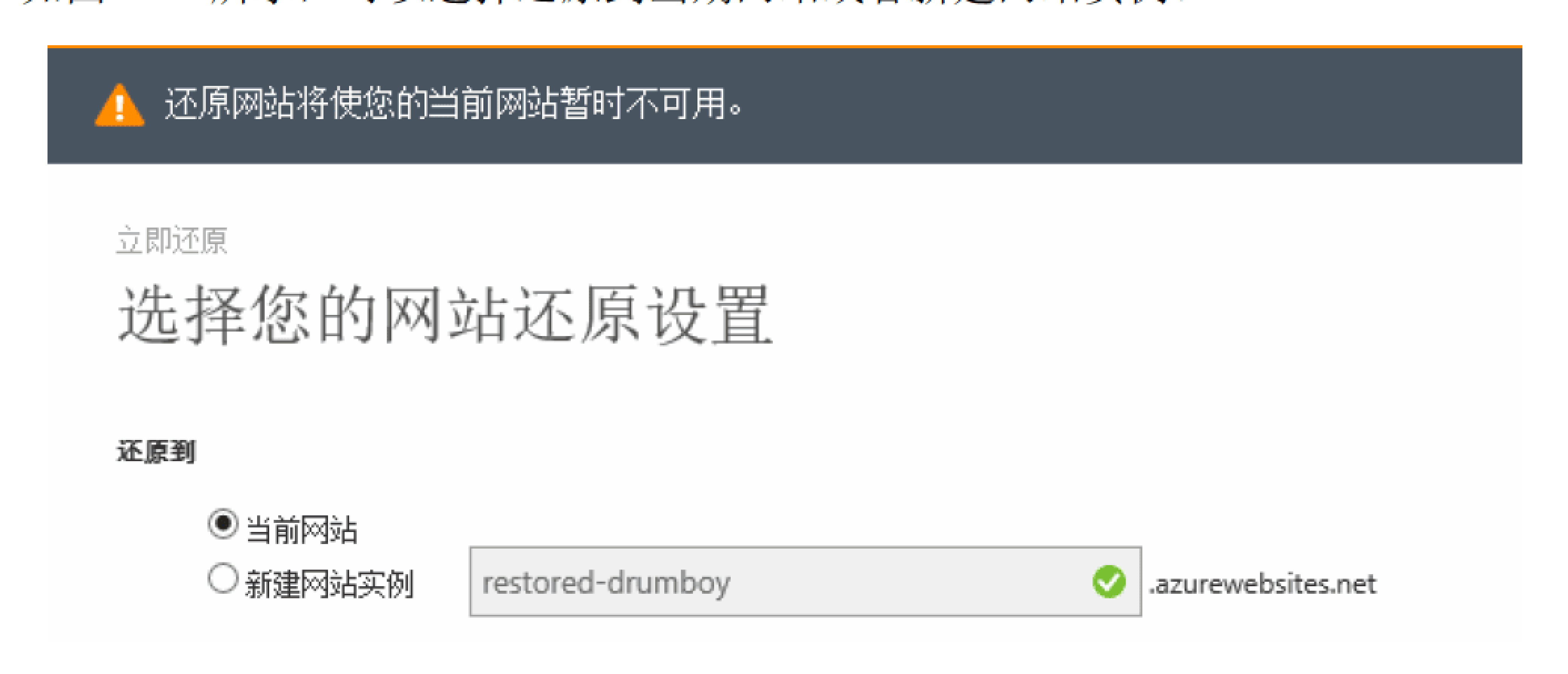


图 2-21 网站还原设置

如果选择“当前网站”，当前网站将被选择的的备份覆盖（破坏性恢复），当前网站的所有内容和配置都将被永久删除，还原操作无法撤销。在还原操作过程中，当前网站将暂时无法使用，直到还原完成。

如果选择“新建网站实例”，将在同一个区域使用指定的名称创建一个新的网站（默认情况下，新的网站名为 resotred-oldWebSiteName）。

恢复后的网站与备份的原始网站具有相同的内容和配置。它还将包括在下面要包括的任何数据库。

如图 2-22 所示，可以选择同时恢复网站数据库。在“包含的数据库”部分，可以指定数据库还原选项：不还原、还原到当前网站相关联的数据库或者新建数据库。

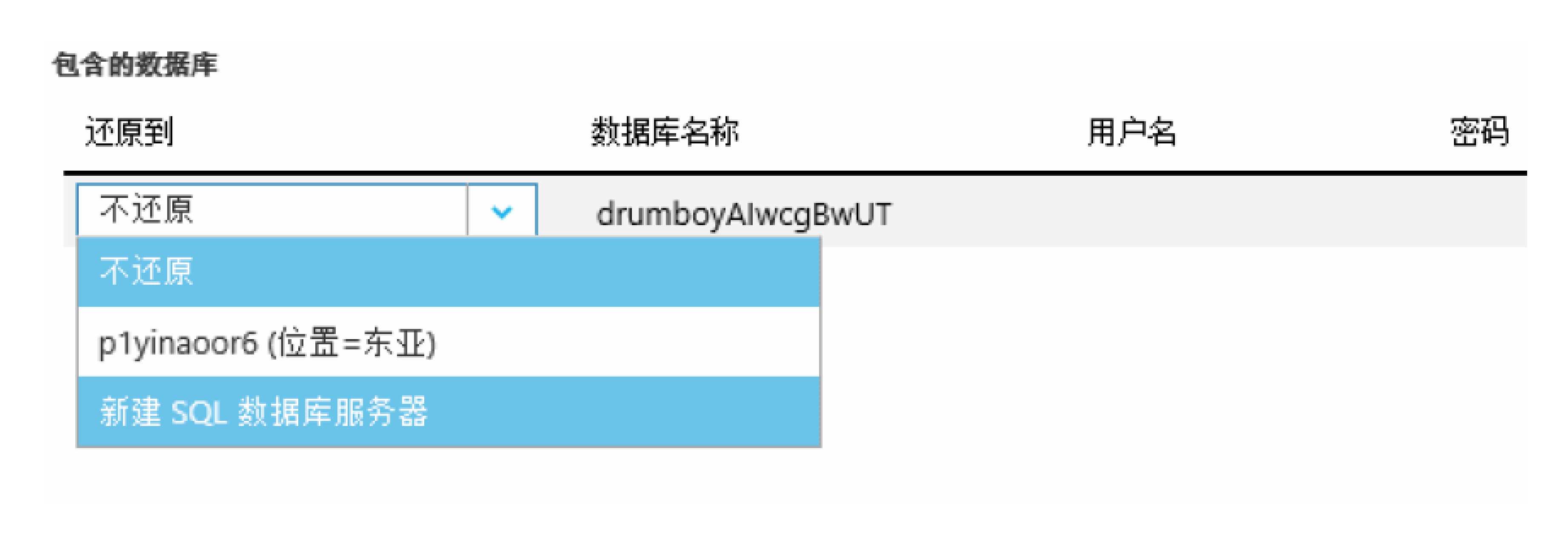


图 2-22 数据库还原设置

注意：如果指定的 SQL 数据库服务器上面已经有相同名称的数据库，必须重新指定一个不同的数据库名称或不同的 SQL Server 服务器。

使用 MySQL 数据库，即使当前 MySQL 数据库服务器已经有相同名称的数据库，仍然可以选择将数据恢复到该服务器。但是，这将清除 MySQL 数据库中的现有内容。

同时可以选择“自动调整连接字符串”来更新网站使用的连接字符串。更新后的连接字符串将指向新的数据库。

（4）在恢复操作结束后，应测试并确认相关的功能在还原完成之后能够正常工作。

2.5　自定义域名

2.5.1　Azure 网站 DNS 简介

创建 Azure 网站时，Azure 自动注册一条网站 DNS 记录。该 DNS 记录将网站名称指向 Azure 网站部署单元的 VIP 地址。在网站创建完成后，可以立即通过 http://*sitename*.azurewebsites.net 来访问您的网站。

注意：每个 Azure 网站部署单元拥有一个固定的 VIP 地址。在每个数据中心，根据客户数量，Azure 会部署一个或者多个部署单元。VIP 地址可以在 Azure 管理门户网站中打开管理域名页面查看。

图 2-23 是作者在香港数据中心创建的两个网站 drumboy 和 debugchina，它们的 DNS 记录都指向香港数据中心的 Azure 网站部署单元的 VIP 地址。

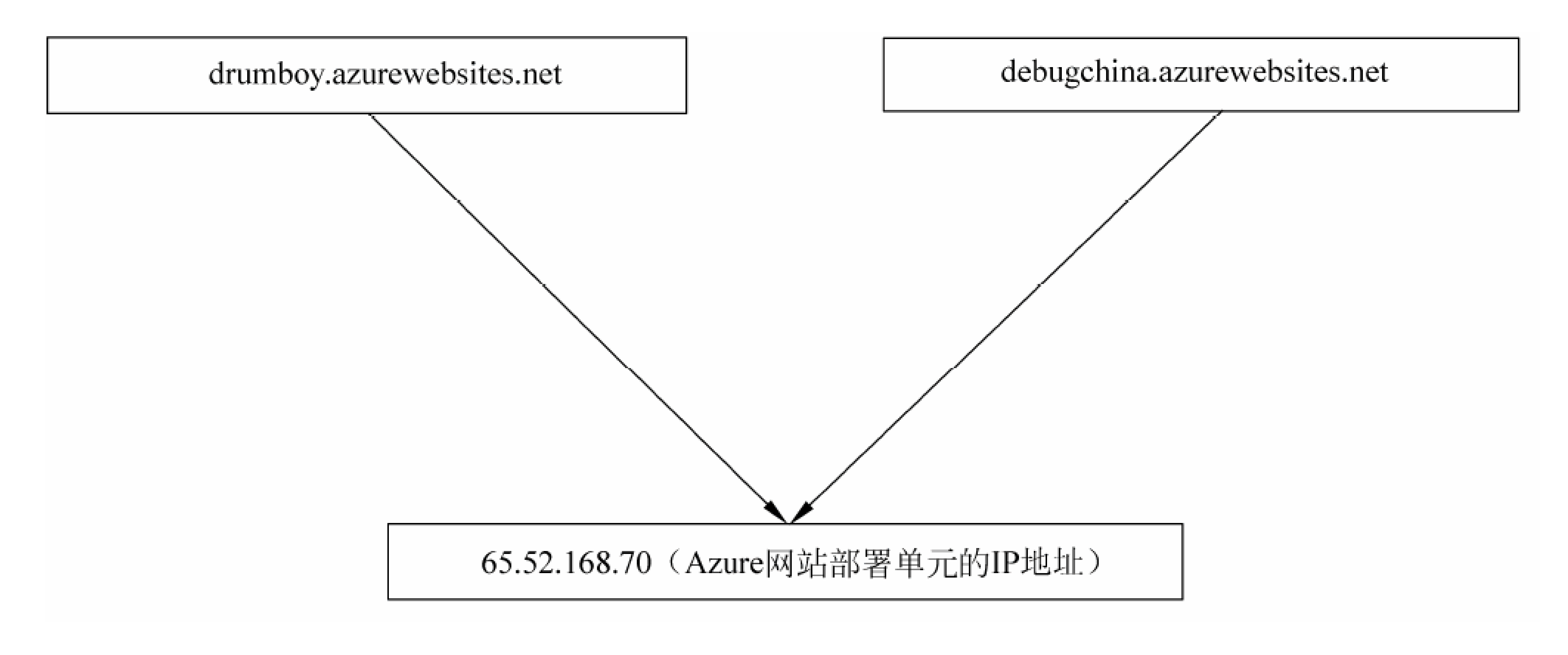


图 2-23　Azure 网站 DNS 设置

利用 Windows 提供的 nskookup 命令行工具，可以更清晰地看到两个网站的 DNS 记录解析结果：

```
C:\Windows\system32>nslookup drumboy.azurewebsites.net
Non-authoritative answer:
Name:     waws-prod-hk1-001.cloudapp.net
Address:  65.52.168.70
Aliases:  drumboy.azurewebsites.net
          waws-prod-hk1-001.vip.azurewebsites.windows.net

C:\Windows\system32>nslookup debugchina.azurewebsites.net
Non-authoritative answer:
Name:     waws-prod-hk1-001.cloudapp.net
Address:  65.52.168.70
```

```
Aliases:  debugchina.azurewebsites.net
          waws-prod-hk1-001.vip.azurewebsites.windows.net
```

2.5.2 配置自有域名

出于商业形象的考虑，商业用户更希望使用自有域名，而不是 azurewebsites.net（或者 ChinaCloudSites.cn）域名。共享模式、基本模式和标准模式网站支持绑定客户自有的域名。客户可以绑定多个自有域名到同一个 Azure 网站。Azure 网站支持客户绑定 3 种类型的域名：

（1）普通域名，比如 www.contoso.com.cn，www.drumboy.cn，blog.drumboy.cn。

（2）裸域名，比如 contoso.com。

（3）通配符域名，比如*.contoso.com。

很多用户对于 DNS 以及域名解析不是很理解，在绑定自有域名时常常遇到很多问题。下面通过实例具体演示如何绑定自有域名到 Azure 网站。绑定自有域名的主要步骤如下：

（1）在绑定自有域名之前，需要向域名注册商申请并注册域名。

（2）申请域名后，需要通过域名注册商开通域名解析。

（3）配置相关 DNS 记录。

（4）绑定到 Azure 网站。

下面通过 drumboy.azurewebsites.net 网站详细演示绑定自有域名的步骤。在绑定自有域名之前，拥有如下资源：

- 在 Microsoft Azure 香港数据中心创建了 drumboy.azurewebsites.net 网站。
- 域名 drumboy.cn。

下面具体演示 3 种常见的自有域名绑定场景。完成后，可以通过下面的域名访问 Azure 网站 drumboy.azurewebsites.net：

http://www.drumboy.cn

http://blog.drumboy.cn

http://drumboy.cn

http://drumboy.azurewebsites.net

2.5.2.1 场景 1：采用 A 记录配置普通域名

在本节，使用 A 记录将 www.drumboy.cn 绑定到 drumboy.azurewebsites.net。绑定后，可以通过 www.drumboy.cn 来访问 Azure 网站。

（1）创建 DNS 授权记录

在绑定自有域名之前，Microsoft Azure 必须确认用户获得将该域名配置为指向 Microsoft Azure 网站的授权。Microsoft Azure 要求用户在 DNS 上配置一条 awverify 的 CNAME 验证记录来确认用户拥有该域名。awverify 表示 Azure Websites Verify（Azure 网站验证）。

以 drumboy 为例，如表 2-1 所示，需要通过域名注册商创建 CNAME DNS 记录。

表 2-1　DNS 授权记录

域　　名	DNS 记录类型	指　　向
awverify.www.drumboy.cn	CNAME	awverify.drumboy.azurewebsites.net

（2）创建 A 记录将自有域名指向 Azure Websites 的 VIP 地址。

登录到管理门户网站，在 drumboy 网站的“配置”页面，定位到“域名”部分，单击“管理域名”，打开“管理自定义域”对话框，会看到 A 记录信息，如图 2-24 所示。

在配置 A 记录时要使用的 IP 地址

65.52.168.70

图 2-24　A 记录信息

注意：每个 Azure 网站部署单元拥有一个固定的 VIP 地址。在每个数据中心，微软会部署一个或者多个部署单元。因此，一定要打开“管理域名”页面确认网站的 VIP 地址。

表 2-2 为通过 DNS 服务提供商创建的 A 记录。

表 2-2　自有域名的 A 记录

域　　名	DNS 记录类型	指　　向
www.drumboy.cn	A 记录	65.52.168.70

（3）等待 DNS 传播，通常需要几分钟到几个小时，极端情况下可能需要 48 小时。

（4）等 DNS 配置传播完成后，即可绑定自有域名。

登录到管理门户网站，在 drumboy 网站的“配置”页面，定位到“域名”部分，单击“管理域名”，打开“管理自定义域”对话框，如图 2-25 所示。输入自有域名 www.drumboy.cn，等待几秒钟，后台完成验证后即可看到绿色的对号。单击“确定”即可完成自有域名绑定。

（5）完成后，在管理门户网站，域名已经被绑定，如图 2-26 所示。

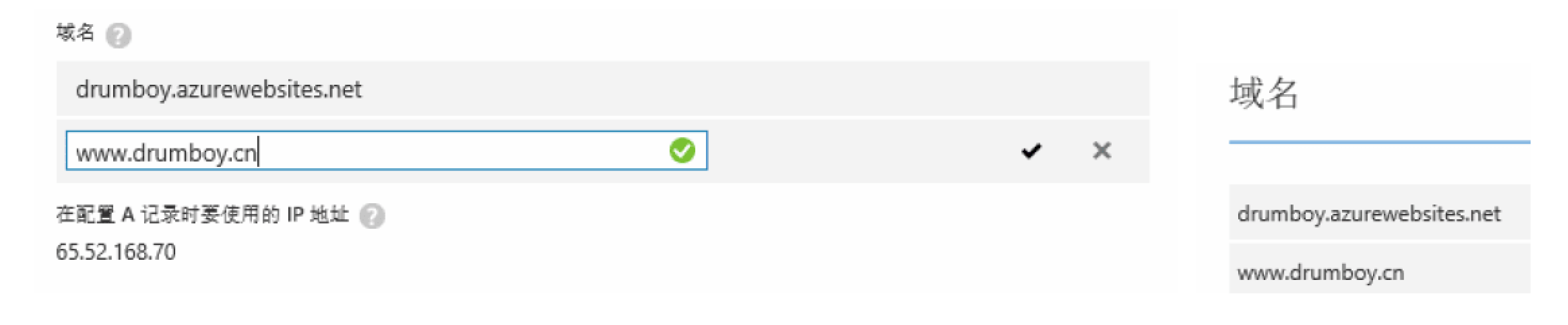


图 2-25　绑定自有域名　　图 2-26　绑定自有域名

（6）至此自定义域名配置成功，网站的客户可以通过 http://www.drumboy.cn 来访问网站。

2.5.2.2　场景 2：采用 CNAME 配置普通域名

在本节使用 CNAME DNS 记录将自有域名 blog.drumboy.cn 绑定到网站 drumboy.azurewebsites.net。绑定后，可以通过 http://blog.drumboy.cn 来访问 Azure 网站。

（1）创建 CNAME 记录。与场景 1 相比，这里并不需要创建 awverify 的授权记录，因

为 CNAME 本身即可作为授权验证信息。如表 2-3 所示，通过 DNS 提供商创建 CNAME 记录。

表 2-3 CNAME DNS 记录

域　　名	DNS 记录类型	指　　向
blog.drumboy.cn	CNAME	drumboy.azurewebsites.net

（2）等待 DNS 传播，通常需要几分钟到几个小时，极端情况下可能需要 48 小时。

（3）等 DNS 配置传播完成后，即可绑定自有域名。

登录到管理门户网站，在 drumboy 网站的“配置”页面，定位到“域名”部分，单击“管理域名”，打开“管理自定义域”对话框，如图 2-27 所示。输入自有域名 blog.drumboy.cn，等待几秒钟完成验证后即可看到绿色的对号。单击“确定”即可完成自有域名绑定。

（4）完成后，在管理门户网站，域名已经被绑定，如图 2-28 所示。

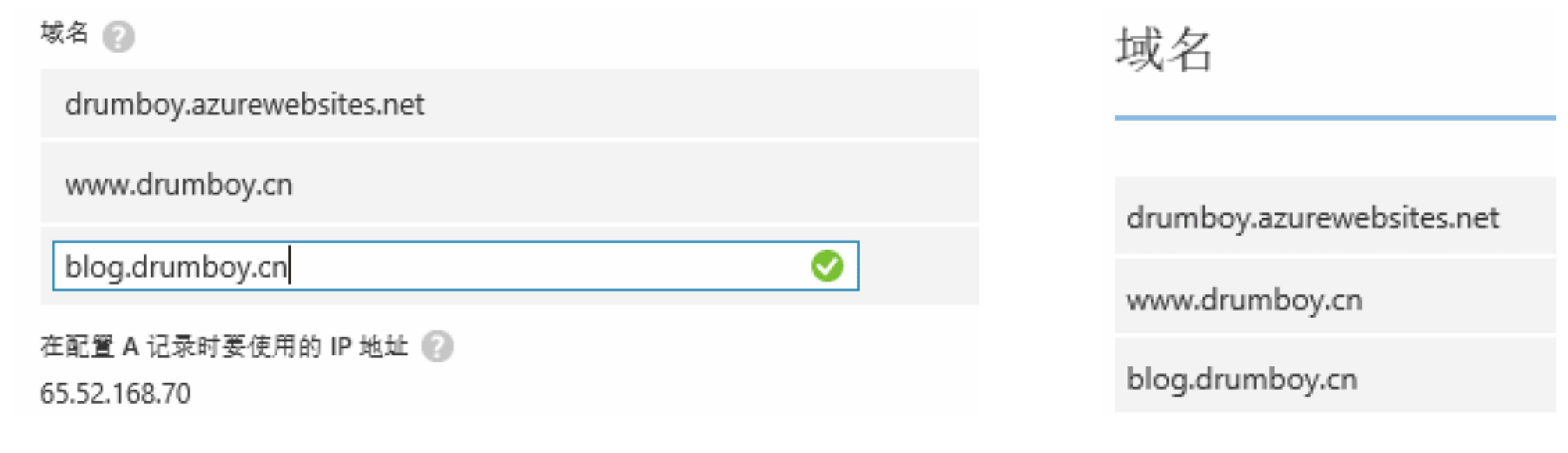


图 2-27 绑定自有域名　　　　图 2-28 绑定自有域名

（5）至此自有域名配置成功，客户可以通过 http://blog.drumboy.cn 访问网站。

2.5.2.3 场景 3：配置裸域名

通过 A 记录将顶级域名（裸域名）drumboy.cn 绑定到 drumboy.azurewebsites.net。绑定后，可以通过 drumboy.cn 来访问 Azure 网站。

（1）创建 DNS 授权记录。与场景 1 相同，这里需要在 DNS 上创建一条 awverify 的 CNAME 记录。以 drumboy.cn 为例，如表 2-4 所示，需要通过 DNS 提供商创建 CNAME 记录。

表 2-4 DNS 授权记录

域　　名	DNS 记录类型	指　　向
awverify.drumboy.cn	CNAME	awverify.drumboy.azurewebsites.net

（2）创建 A 记录将自有域名指向前端服务器 IP 地址。

登录到管理门户网站，在 drumboy 网站的“配置”页面，定位到“域名”部分，单击“管理域名”，打开“管理自定义域”对话框，会看到如图 2-29 所示的 A 记录信息。

在配置 A 记录时要使用的 IP 地址

65.52.168.70

图 2-29 配置 A 记录使用的 IP 地址

如表 2-5 所示，需要通过 DNS 服务提供商创建 A 记录。

表 2-5　CNAME DNS 记录

域　　名	DNS 记录类型	指　　向
Drumboy.cn	CNAME	65.52.168.70

（3）等待 DNS 传播，通常需要几分钟到几个小时，极端情况下可能需要 48 小时。

（4）等 DNS 配置传播完成后，即可绑定自有域名。

登录到管理门户网站，在 drumboy 网站的“配置”页面，定位到“域名”部分，单击“管理域名”，打开“管理自定义域”对话框，如图 2-30 所示。输入自有域名 www.drumboy.cn，等待几秒钟完成验证后即可看到绿色的对号。单击“确定”即可完成自有域名绑定。

（5）完成后，在管理门户网站，域名已经被绑定，如图 2-31 所示。

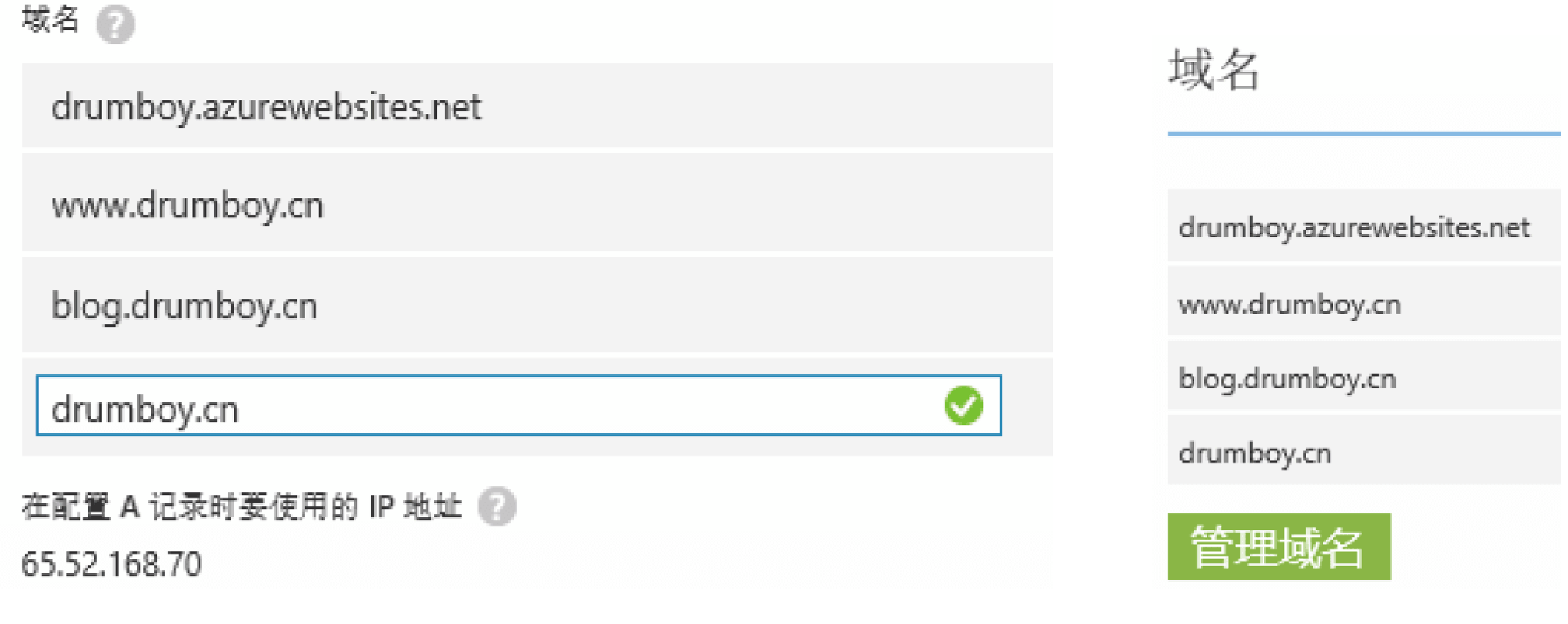


图 2-30　绑定裸域名　　　图 2-31　裸域名绑定结果

（6）绑定裸域名后，客户可以通过 http://drumboy.cn 访问网站。

2.5.3　深入 Azure 网站自有域名配置

通过上面的 3 种场景，可以看到 Microsoft Azure 网站可以通过 A 记录或者 CNAME 记录来配置自有域名。使用 A 记录的时候，需要配置 DNS 授权记录（awverify）。如果需要配置顶级域名（裸域名，比如 drumboy.cn），则只能使用 A 记录。

微软公司保证 Azure 网站部署单元使用的 IP 地址永远不会改变。尽管如此，基于下面的原因，推荐使用 CNAME。

（1）CNAME 配置相对容易。

（2）如果将网站从一个数据中心迁移到另外一个数据中心，在使用 CNAME 的情况下，不需要任何的 DNS 设置修改。但是，在使用 A 记录的情况下，由于每个数据中心的 Azure 网站的 VIP 地址不同，需要通过 DNS 提供商修改 A 记录的配置。

（3）后面会介绍到，网站启用 IP SSL 后，网站 IP 地址会变化。同上，如果采用 A 记录，则需要修改 DNS 配置，采用 CNAME 不需要修改 DNS 配置。

在配置 DNS 之前，需要确保 DNS 已经传播成功。通常可以通过一些第三方的 DNS 查询工具来进行，例如使用 Windows 提供的 NSLOOKUP 命令和如下网站进行查询：

http://www.digwebinterface.com

http://querydns.org

通过 DNS 查询工具，可以清楚地发现 Azure 网站的 DNS 具体配置（以香港数据中心部署单元 001 为例），如图 2-32 所示。

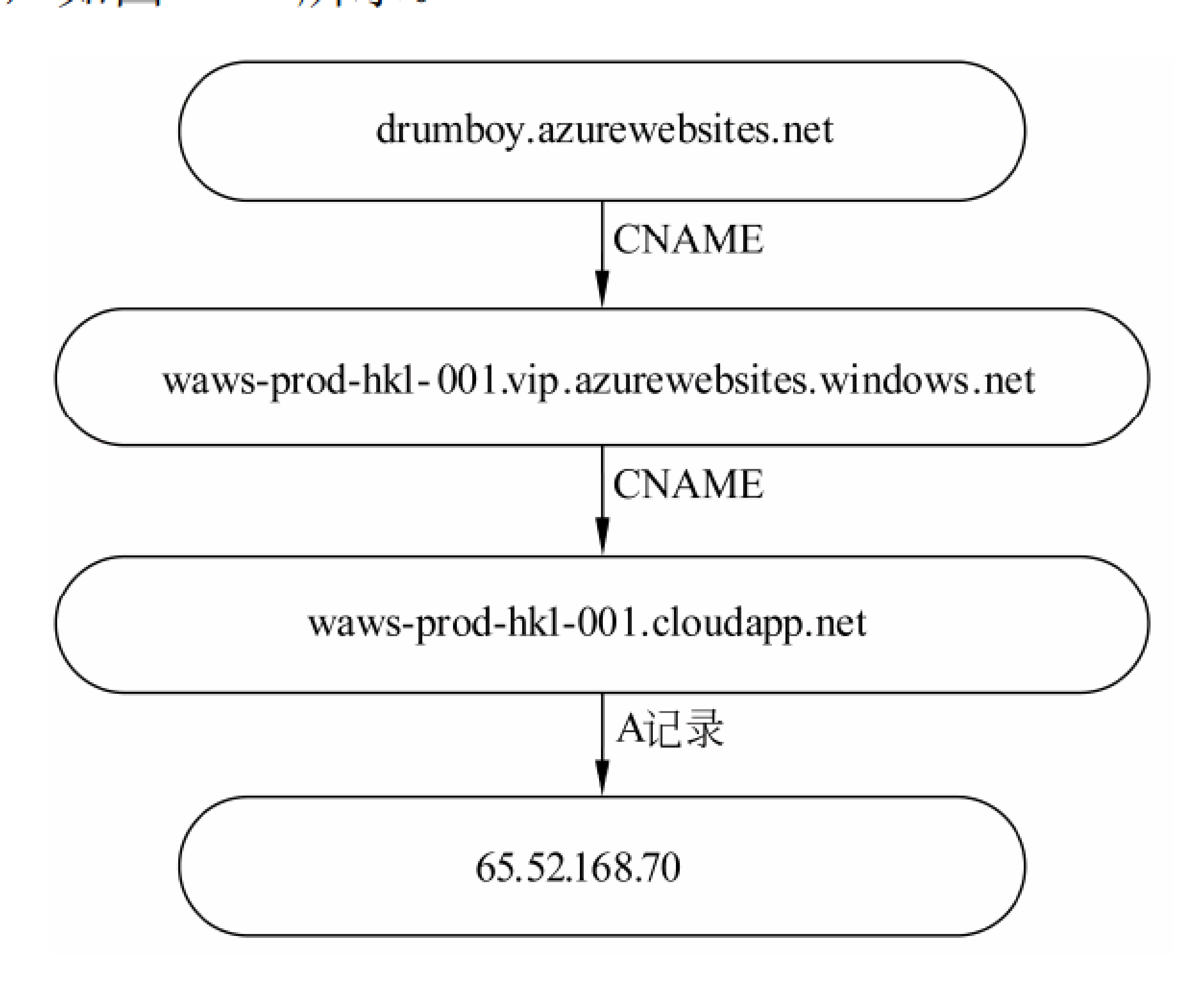


图 2-32　Azure 网站 DNS 配置

2.5.4　Azure 网站 DNS 配置检查清单

2.5.2 节演示了如何将自有域名绑定到 Azure 网站。本节总结 3 种不同类型的域名绑定到 Azure 网站需要的 DNS 配置清单。

2.5.4.1　普通域名

普通域名即常见的三段式域名，比如 www.microsoft.com、www.contoso.com、blog.mydomain.com 等，也包括 www.vikram.com.cn、products.myCorp.com.cn 等域名。

如同前面所示，Azure 网站支持使用 A 记录或者 CNAME 方法配置三级域名。

1. A 记录

以 www.contoso.com 为例，将 www.contoso.com 绑定到 Azure 网站 mySite.azurewebsites.net。在管理门户网站，配置 DNS 界面显示的 IP 地址为 65.52.168.70。采用 A 记录的方式需要配置如表 2-6 所示的 DNS 记录。

表 2-6　普通域名采用 A 记录的 DNS 配置清单

域　　名	DNS 记录类型	DNS 指向
www. contoso.com	A 记录	65.52.168.70（显示在管理门户的 IP 地址）
Awverify.www.contoso.com	CNAME	Awverify.mySite.azurewebsite.net

2. CNAME

采用 CNAME 方式，只需配置如表 2-7 所示的 DNS 记录。

表 2-7　普通域名采用 CNAME 记录的 DNS 配置清单

域　　名	DNS 记录类型	DNS 指向
www.contoso.com	CNAME	mySite.azurewebsites.net

2.5.4.2　裸域名

myCompany.com、Microsoft.com、Contoso.com、GitHub.com 等都属于裸域名，也包括像 vikram.com.cn 等域名。Azure 网站支持绑定裸域名。因为 DNS 协议不支持裸域名配置 CNAME 记录，所以配置裸域名只能通过 A 记录的方式。DNS 记录的配置如表 2-8 所示。

表 2-8　裸域名配置的 DNS 配置清单

域　　名	DNS 记录类型	DNS 指向
contoso.com	A 记录	65.52.168.70（显示在管理门户的 VIP 地址）
Awverify.mydomain.com	CNAME	Awverify.mySite.azurewebsite.net

2.5.4.3　通配符域名

如果配置了通配符域名记录，DNS 服务器在没有匹配到对应记录时，将返回通配符记录。通配符 DNS 记录使用“*”作为最左边的域名标识符，如*.mycompany.com、*.contoso.com、*.vikram.com.cn 等域名。

当 DNS 服务器收到 DNS 查询请求时，比如查询 abc.contoso.com，DNS 服务器首先检查是否有匹配的 abc 记录。如果没有匹配的 abc 记录，继续检查是否有通配符记录。如果查询到通配符记录则返回通配符域名，否则返回域名不存在错误。

Azure 网站支持绑定通配符域名。绑定通配符域名只能使用 CNAME 记录，不支持 A 记录。要绑定通配符域名到 Azure 网站，需要按表 2-9 所示配置 DNS 记录。

表 2-9　通配符域名配置的 DNS 配置清单

域　　名	DNS 记录类型	DNS 指向
*.mydomain.com	CNAME	myAzureSite.azurewebsites.net

2.5.5　绑定自有域名后的 DNS 配置

图 2-33 描述了配置客户自有域名后的 DNS 架构。

2.5.6　中国区 Azure 网站 DNS 配置

以上所有信息都适用于中国区 Azure 网站。唯一的区别就是中国区 Azure 网站的域名后缀为 ChinaCloudSites.cn。

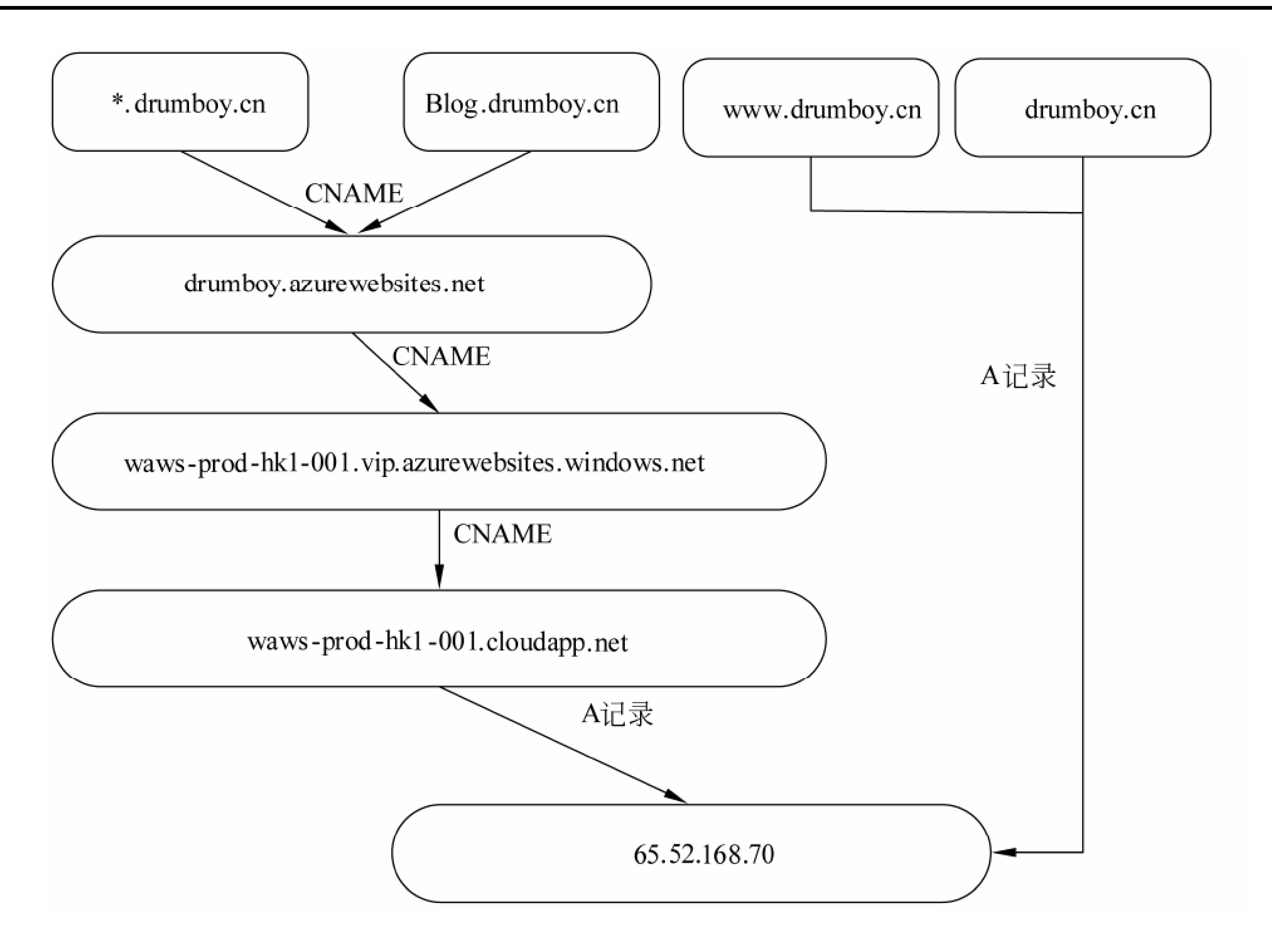


图 2-33　绑定自有域名后的 DNS 配置

2.6　配置 SSL 绑定

Microsoft Azure 网站内建支持两种 HTTPS 方式以满足客户应用的安全需求：SNI SSL 和 IP SSL。

如图 2-34 所示，Azure 网站采用 SSL（off load）卸载功能。客户端到前端服务器使用 HTTPS 建立安全连接。前端服务器到工作服务器使用 HTTP。目前暂时不支持采用客户端证书进行双向加密认证。

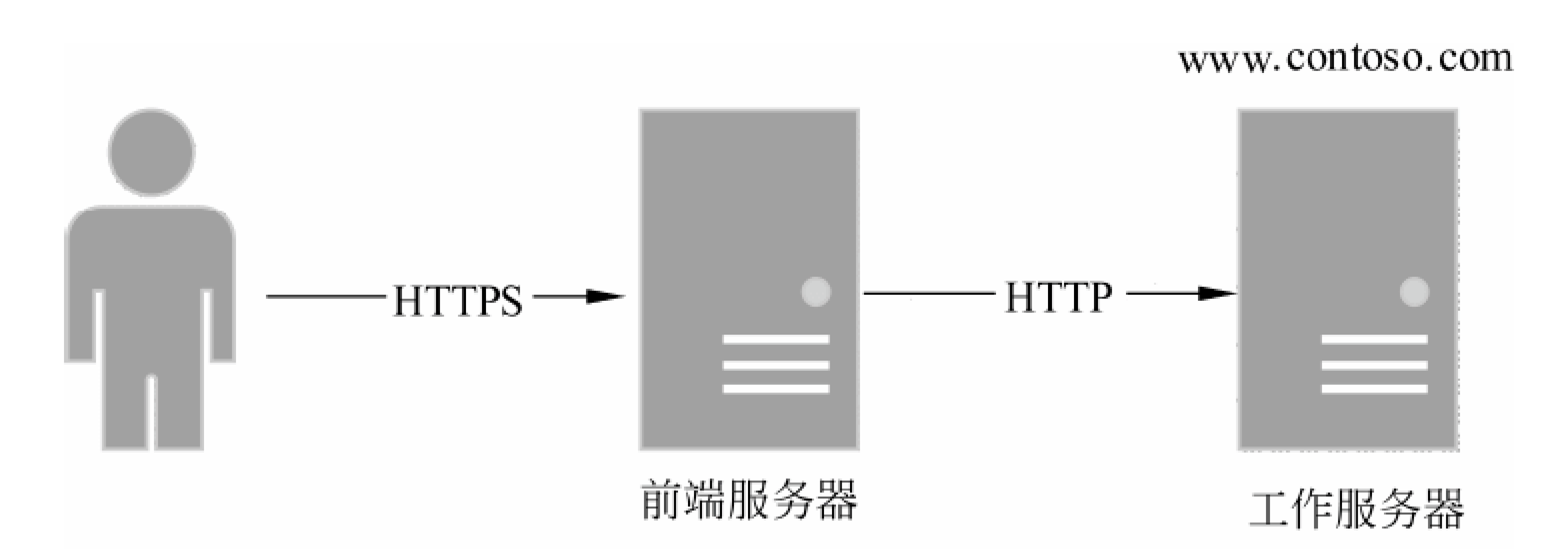


图 2-34　Azure 网站 SSL 工作模式

在 Azure 网站前端服务器默认安装了*.azurewebsites.net 的证书。客户网站创建后，即可使用 HTTPS://<sitename>.azurewebsites.net 访问，此时采用的是*.azurewebsites.net 证书。

出于商业机密以及商业形象的考虑，商业客户更希望使用自有域名和自有安全证书，比如 https://www.contoso.com。只有基本模式和标准模式下的网站支持客户使用自有安全证书。免费模式和共享模式网站只能使用默认安装的*.azurewebsites.net 证书。

2.6.1　Azure 网站 SSL 绑定模式

下面首先介绍 SNI SSL 和 IP SSL 两种模式的区别，然后具体演示如何配置 SSL 证书。

2.6.1.1　SNI SSL

采用传统的 SSL 安全连接，服务器端采用“IP 地址+TCP 端口号”作为唯一的标识。在这种情况下，如果网站要求采用标准的 443 端口，那么每个 IP 地址只能支持一个 HTTPS 网站。如果服务器只有一个 IP 地址，那么它只能支持一个网站采用 HTTPS。

IIS 6 及后续版本通过支持通配符证书和 SAN 证书允许在一个 IP 地址上运行多个 HTTPS 网站。但是，这有很大的局限。比如，如果通配符证书是颁发给*.microsoft.com（CN=*.microsoft.com）。那么，要求所有的网站域名后缀必须是 microsoft.com。例如 support.microsoft.com、msdn.microsoft.com 等。

SNI（Server Name Indication）是 TLS 协议的一个扩展。通过 SNI，即使只有一个 IP 地址，也可以同时运行多个支持 HTTPS 的不同域名的网站，而且这些网站可以使用不同的证书。IIS 8 及后续版本支持 SNI，可以阅读下面的文档深入了解 IIS SNI 的功能：

http://www.iis.net/learn/get-started/whats-new-in-iis-8/iis-80-server-name-indication-sni-ssl-scalability

注意：SNI 需要浏览器的支持。一些老的浏览器并不支持 SNI 功能。比如，所有运行在 Windows XP 上的浏览器都不支持 SNI。如果 Windows XP 的用户使用 https://www.contoso.com 访问网站，客户会发现使用的并不是网站的自有证书，而是*.azurewebsites.net 的证书。如果业务需要支持 Windows XP 的用户使用 https://www.contoso.com 访问网站，则需要使用基于 IP 的 SSL。

Azure 网站采用了 IIS 8 的 CCS（Central Certificate Store）功能实现 SNI SSL。可以阅读下面的文章深入了解 IIS 8 的 CCS 功能：

http://www.iis.net/learn/get-started/whats-new-in-iis-8/iis-80-centralized-ssl-certificate-support-ssl-scalability-and-manageability

2.6.1.2　IP SSL

大多数情况下，SNI SSL 即可满足需求。如果网站有很多客户仍然使用不支持 SNI 的浏览器，那么需要考虑采用 IP SSL 功能。

当网站启用 IP SSL 后，Azure 为网站分配一个专用的公网 IP 地址。需要将网站的自有域名绑定到分配给网站的公网 IP 地址。如图 2-35 所示，可以在 Microsoft Azure 管理门户网站的“仪表盘”选项卡上的“速览”下面找到分配给网站的 IP 地址。

虚拟 IP 地址
65.52.170.151

图 2-35 IP SSL 的专有公网 IP 地址

注意：

（1）每个标准模式的网站只能有一个专有 IP 地址。

（2）每个标准的托管计划已经包含了一个免费的专有 IP 地址。

（3）如果同一个托管计划下，两个标准模式网站同时采用 IP SSL，则每个网站分配一个专有 IP 地址。其中一个免费，另外一个则需要付费。

（4）一旦停用 IP SSL 后，分配给网站的 IP 地址立即被释放。当再次启用 IP SSL 时，可能会被分配一个不同的 IP 地址。如果采用 A 记录配置网站的自有域名，则需要重新配置 DNS。建议采用 CNAME 记录配置 DNS。

2.6.2 配置安全证书

可以通过 Azure 管理门户网站、Powershell 和 X-CLI 命令行配置安全证书。通过门户网站配置安全证书非常简单。但是，很多客户因为缺乏对 SNI SSL、IP SSL 和证书的了解，遇到许多问题。下面通过实际的例子具体讲解如何配置 SSL。

2.6.2.1 生成证书

商业应用需要向证书颁发机构申请一个证书。关于如何申请证书请联系证书供应商。

Microsoft Azure 网站支持 3 种证书：

（1）基本证书。该基本证书的通用名称（CN）设置为特定的域名或子域名。客户端将使用指定的域名来访问该网站，例如 www.contoso.com。这些证书只能用于指定的单个域名。

（2）通配符证书。该证书的 CN 在子域名级别包含通配符“*”。通配符证书可以用于任何指定的子域名。例如，*.contoso.com 可以用于 www.contoso.com、payment. contoso.com 和 login.contoso.com。但是它不能用于 test.login.contoso.com，因为这增加了一层额外的子域。它也不能用于 contoso.com，因为这是在根域级别，而不是一个子域。Microsoft Azure 网站自动提供一个通配符证书*.azurewebsites.net。

（3）SAN 证书。SAN（Subject Alternative Name）是一个证书扩展，它允许额外的替代名称。例如，一个 SAN 证书可具有 contoso.com 的 CN，除此之外还可以有 www.contoso.com、payment.contoso.com、test.login.contoso.com 的替代名称，甚至 www.foo.com。这种证书适用于通用名称和替代名称指定的所有域名。

下面使用一个自签名证书来演示如何为 www.drumboy.cn 配置 SNI SSL 和 IP SSL。生成自签名证书的方式有很多，比如 makecert 工具、IIS 管理器、selfssl 工具和 OpenSSL 等。

IIS 管理器生成的自签名证书通用名称总是指向生成证书的机器名，所以不能用于 Azure 网站。PowerShell 中的 New-SelfSignedCertificate 不兼容 Azure 网站，也不能使用。下面使用 makecert 工具来生成自签名证书。

（1）首先以管理员身份运行 PowerPhell 控制台或者 PowerShell ISE，命令行控制台也可。

（2）在 PowerShell 控制台运行下面的命令：

```
makecert -r -pe -b 01/01/2014 -e 01/01/2034 -eku 1.3.6.1.5.5.7.3.1 -ss My
-n CN=www.drumboy.cn -sky exchange -sp "Microsoft RSA SChannel Cryptographic
Provider" -sy 12 -len 2048
```

其中 www.drumboy.cn 是要绑定证书的 Azure 网站的域名。该命令将生成通用名称为 www.drumboy.cn 并将证书安装在当期用户的证书库中。

（3）在 PowerShell 控制台运行下面的命令导出证书：

```
$mypwd = ConvertTo-SecureString -String "123654" -Force -AsPlainText
get-childitem cert:\currentuser\my -dnsname www.drumboy.cn | export-
pfxcertificate -filepath c:\www.drumboy.cn.pfx -password $mypwd
```

该命令将前面生成的证书导出并保存到 c:\www.drumboy.cn.pfx，使用密码 123654 保护。下面将使用该证书来 www.drumboy.cn 提供 HTTPS 服务。

2.6.2.2　场景 1：配置 SNI SSL

配置步骤如下：

（1）在浏览器中打开 Microsoft Azure 管理门户网站。

（2）单击网站的名称，然后选择“配置”选项卡。

（3）浏览“证书”部分，选择“上载证书”。

（4）在“上载证书”对话框，选择前面生成的证书 www.drumboy.cn.pfx，并输入证书密码，如图 2-36 所示。单击“确定”上传证书。



图 2-36　上传网站使用的证书

（5）证书上传后，在“证书”部分会显示证书的有效期和指纹信息，如图 2-37 所示。

（6）在“配置”选项卡的“SSL 绑定”部分，使用下拉菜单选择要使用 SSL 保护的域名，并指定刚刚上传的证书，最后选择使用 SNI，如图 2-38 所示。



图 2-37　网站证书信息

图 2-38　SNI 绑定

（7）单击命令栏的“保存”按钮，保存设置。

现在打开浏览器，然后访问 https://www.drumboy.cn，浏览器会提示该网站的安全证书是由不受信任的机构颁发的。选择“继续浏览”，此时浏览器会显示网站内容。单击浏览器导航栏的“证书错误”按钮，选择“查看证书”，此时使用的是 www.drumboy.cn 的证书，而不是*.azurewebsites.net 的证书，如图 2-39 所示。

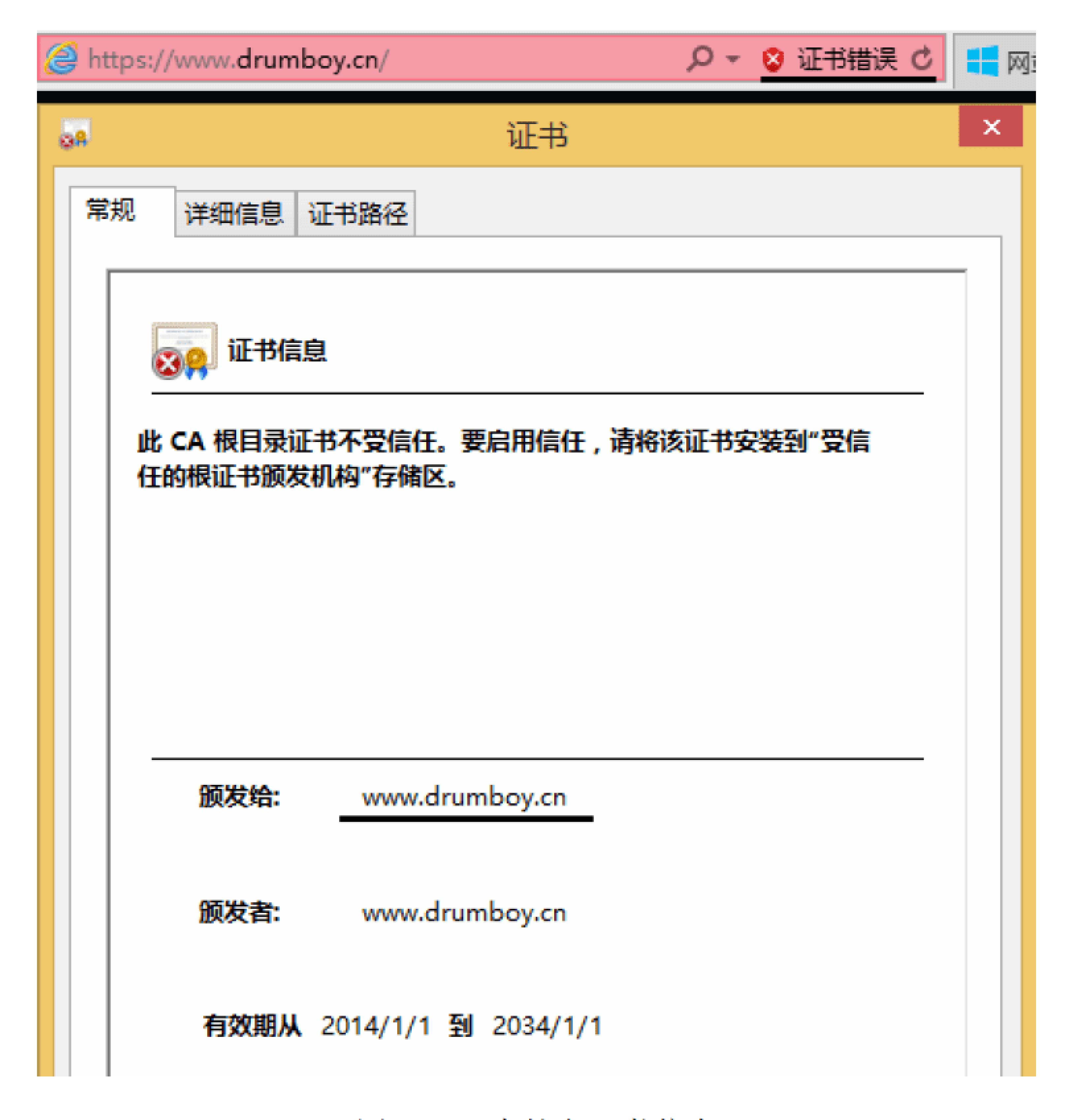


图 2-39　自签名证书信息

注意：看到证书错误是因为该证书是一个自签名证书，而不是受信任的证书机构颁发的证书。将该证书导入本机的“受信任的根证书颁发机构”后，就不会再看到这个错误提示。

2.6.2.3　场景 2：配置 IP SSL

上面已经为网站 www.drumboy.cn 配置好了 SNI SSL，它使用的是自己上传的证书。如果使用一个不支持 SNI 的浏览器，会是什么情景呢？前面讲过，SNI 是 TLS 协议的一个扩展。可以通过禁止 IE 11 浏览器使用 TLS 1.0、1.1、1.2 功能来模拟一个不支持 SNI 的浏览器，如图 2-40 所示。

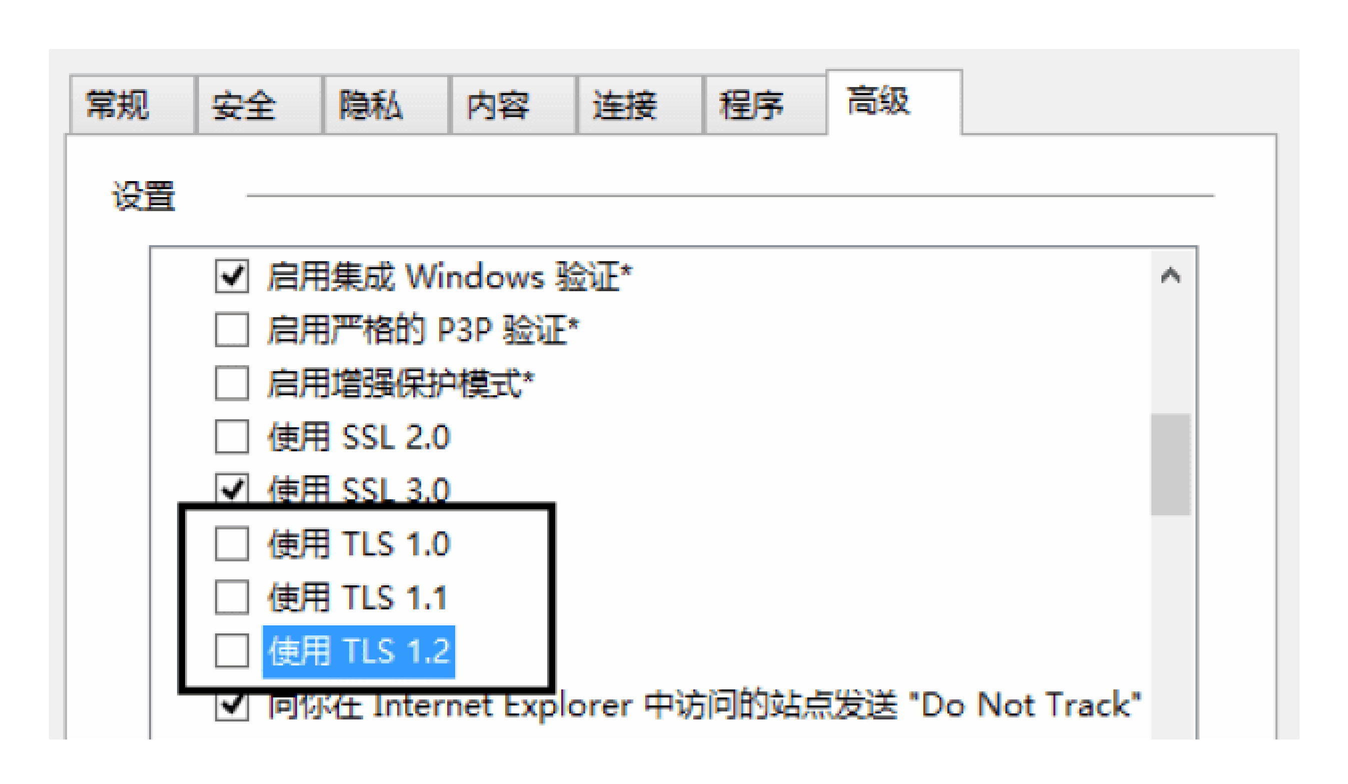


图 2-40　模拟不支持 SNI 的浏览器

此时，再次用浏览器访问 HTTPS://www.drumboy.cn，可以看到这次使用的是 *.azurewebsites.net 的证书，而不是颁发给 www.drumboy.cn 的证书，如图 2-41 所示。

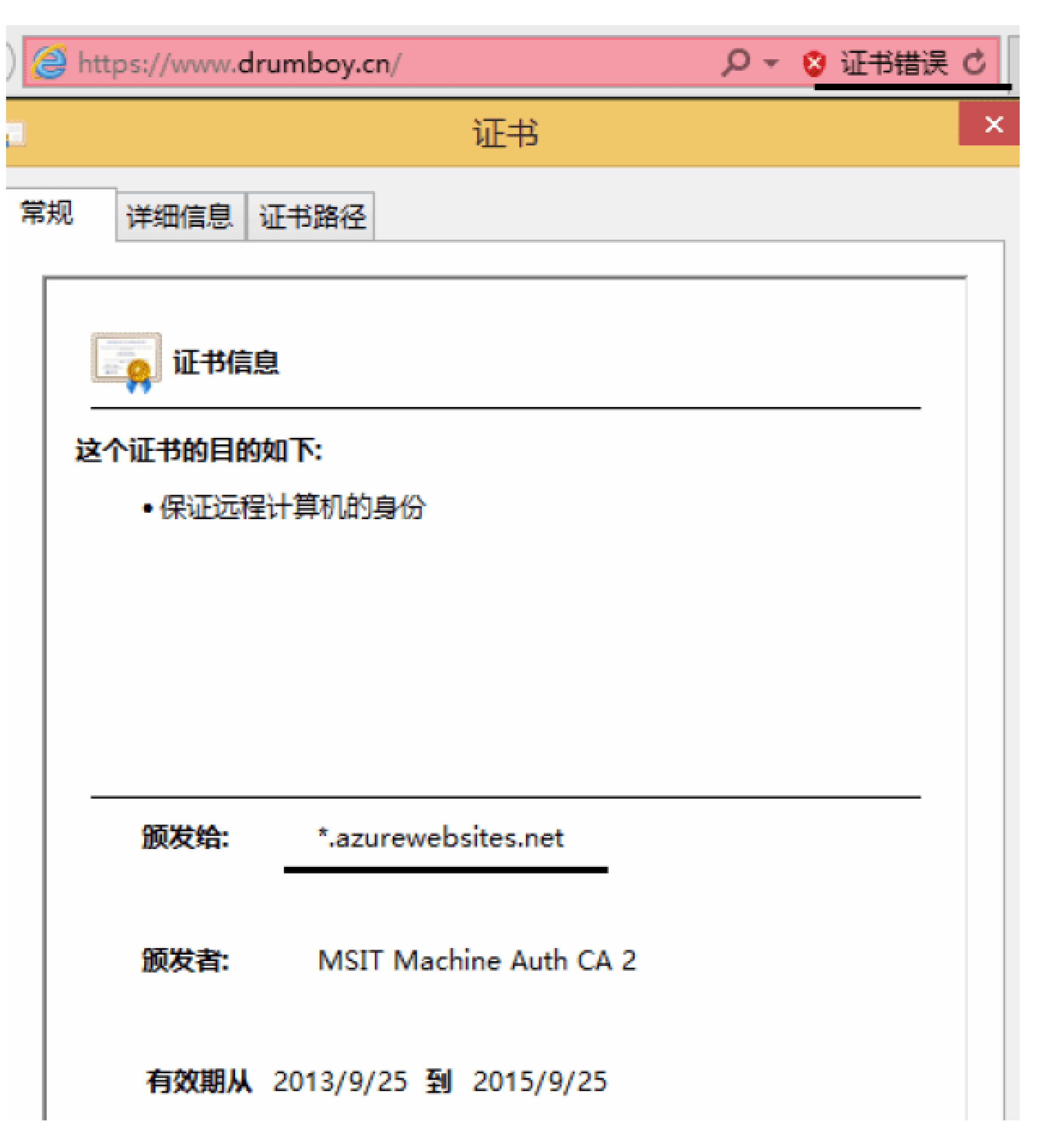


图 2-41　证书信息

如果网站的客户很大一部分采用不支持 SNI 的浏览器，而且商业需求不允许出现证书错误警告，那么需要考虑采用 IP SSL 模式。下面演示配置 IP SSL 绑定模式的步骤。

（1）回到 Microsoft Azure 管理门户网站之前配置 SSL 的部分。

（2）将之前配置的 SNI SS 修改为 IP SSL，如图 2-42 所示。

ssl 绑定

www.drumboy.cn	www.drumboy.cn, 过期: 2034/...	基于 IP 的 SSL
选择域名	选择证书	SNI SSL

图 2-42　配置 IP SSL 绑定

（3）单击页面下方命令栏中的“保存”，保存修改。

（4）单击页面上方的“仪表盘”选项卡，切换到“仪表盘”，在页面右侧“速览”部分，会看到分配给网站的专属地址，如图 2-43 所示。

虚拟 IP 地址

65.52.170.151

图 2-43　网站专属公网 IP

（5）重新配置 DNS。

如表 2-10 所示，如果使用 CNAME 记录配置网站的自有域名，则不需要修改 DNS 配置。这也是建议使用 CNAME 配置自有域名的另外一个原因。

表 2-10　CNAME DNS 记录

域　　名	DNS 记录类型	指　　向
www.drumboy.cn	CNAME 记录	drumboy.azurewebsites.net

如果使用 A 记录来配置网站的自有域名，指向部署单元共享 IP 地址的 A 记录如表 2-11 所示。

表 2-11　指向部署单元共享 IP 地址的 A 记录

域　　名	DNS 记录类型	指　　向
www.drumboy.cn	A 记录	65.52.168.70（共享 IP 地址）

需要将 IP 地址修改为表 2-12 中的 IP SSL 专用 IP 地址。

表 2-12　指向网站专属 IP 地址的 A 记录

域　　名	DNS 记录类型	指　　向
www.drumboy.cn	A 记录	65.52.170.151（IP SSL 专用地址）

图 2-44 描述了当启用 IP SSL 时，如果使用 A 记录配置自有域名时如何修改 DNS。

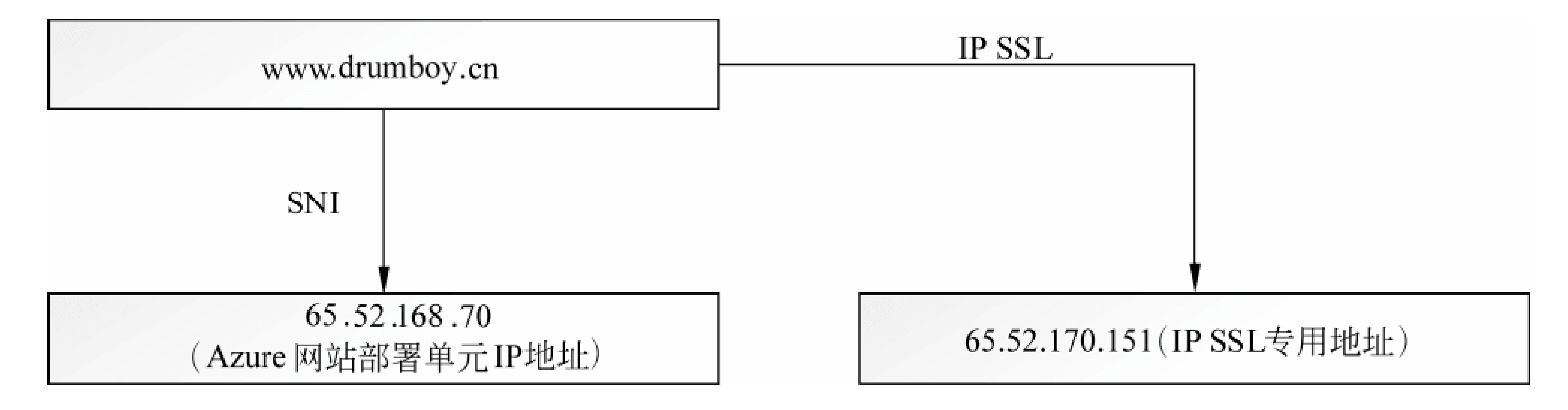


图 2-44　采用 IP SSL 后 IP 地址的变化

此时，客户使用不支持 SNI 的浏览器访问 HTTPS://www.drumboy.cn 时，安全连接采用颁发给 www.drumboy.cn 的证书进行加密，而不是微软公司自动提供的*.azurewebsites.net 证书。

2.6.3　深入 IP SSL DNS 配置

使用 nslookup 命令行，或者借助 www.digwebinterface.com，可以查看 IP SSL 的 DNS 配置。如果 2-45 所示，当启用 IP SSL 时，Azure 网站自动创建 DNS 配置。因此，如果使用 CNAME 来配置网站的自有域名，就不需要重新配置 DNS，在 IP SSL 和 SNI SSL 之间切换也不会导致服务中断。

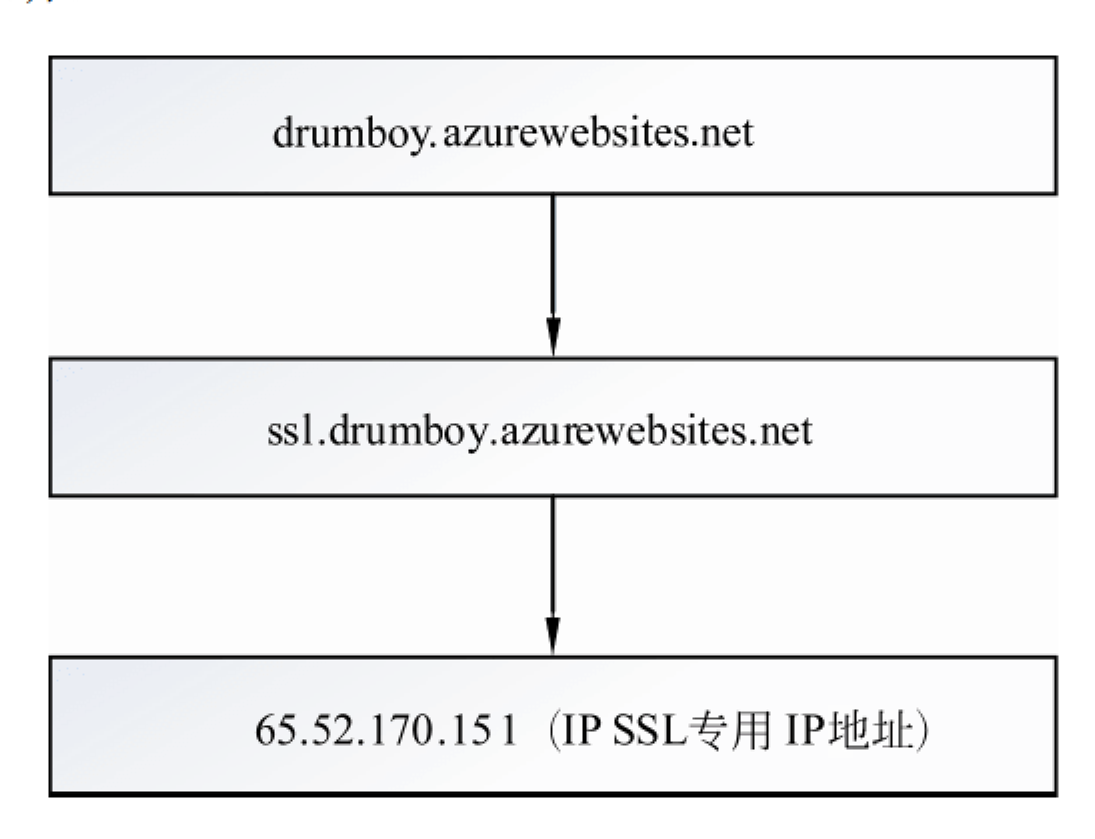


图 2-45　IP SSL 的 DNS 配置

图 2-46 描述了使用 CNAME 配置自有域名时 SNI SSL 和 IP SSL 的 DNS 配置。

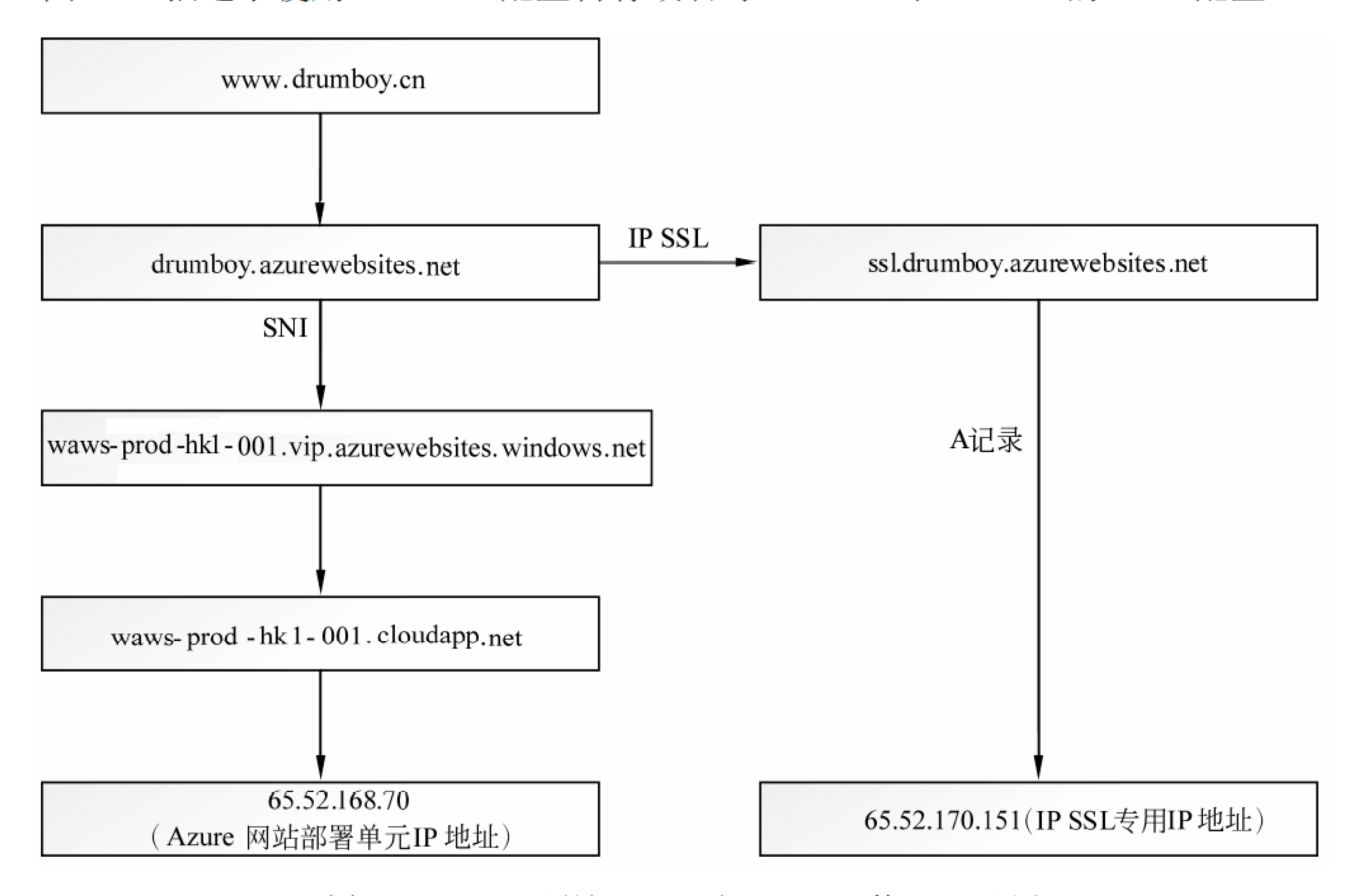


图 2-46　Azure 网站 IP SSL 和 SNI SSL 的 DNS 配置

2.6.4　同时使用 IP SSL 和 SNI SSL

每个标准模式的托管计划包含了一个 IP SSL 绑定和 5 个 SNI 绑定。如果网站绑定了多个自有域名，那么可以同时绑定 IP SSL 和 SNI SSL。

下面通过一个实例来具体演示如何配置 Azure 网站同时使用 IP SSL 和 SNI SSL 绑定。网站 maws.azurewebsites.net 绑定了两个自有域名，分别采用 SNI SSL 和 IP SSL：

- modern.waws.cn（采用 SNI SNI 绑定）。
- legacy.waws.cn（采用 IP SSL 绑定）。

那么应该如何分别配置 DNS 和 SSL 绑定呢？

首先，上传证书到 Azure 网站。关于如何生成测试证书并上传证书到 Azure 网站，请参考前面的内容。如图 2-47 所示，分别上传主题为 modern.waws.cn 的证书和 legacy.waws.cn 的证书到 Azure 网站。

证书

主题	到期日期	指纹
modern.waws.cn	2040/1/1	00239A3E4145CEF79DD471450616818627411739
antarestest.trafficmanager.net	2040/1/1	1C9D492CC4BEFD2EE8F4F0F4FC66117111C55371
wzhaodnn.azurewebsites.net	2040/1/1	75DE53FBD9935BE6A2B4ACB8DCE7C029356C3292
obsolete.waws.cn	2040/1/1	C29C916A8C574FC0B14314C9F154249DA0C7856B
legacy.waws.cn	2040/1/1	D08457616B45DDE805BA59E58F1764C32F098D85

上载证书

图 2-47　证书列表

2.6.4.1　配置 IP SSL 绑定

配置 IP SSL 绑定的步骤如下：

（1）绑定 legacy.waws.cn 的自有域名。

Legacy.waws.cn 采用 IP SSL 绑定，可以配置 CNAME 记录或者 A 记录。以 CNAME 记录为例，如表 2-13 所示，需要在 DNS 服务提供商注册一条 CNAME 记录。

表 2-13　IP SSL 的 CNAME 记录

域　　名	DNS 记录类型	指　　向
Legacy.waws.cn	CNAME 记录	maws.azurewebsites.net

配置完成后，登录到 Azure 管理门户网站，如图 2-48 所示，将自有域名 legacy.waws.cn 绑定到 maws 网站。

（2）配置 IP SSL 绑定。在管理门户网站创建一个 IP SSL 绑定，绑定到主题为 legacy.waws.cn 的证书。

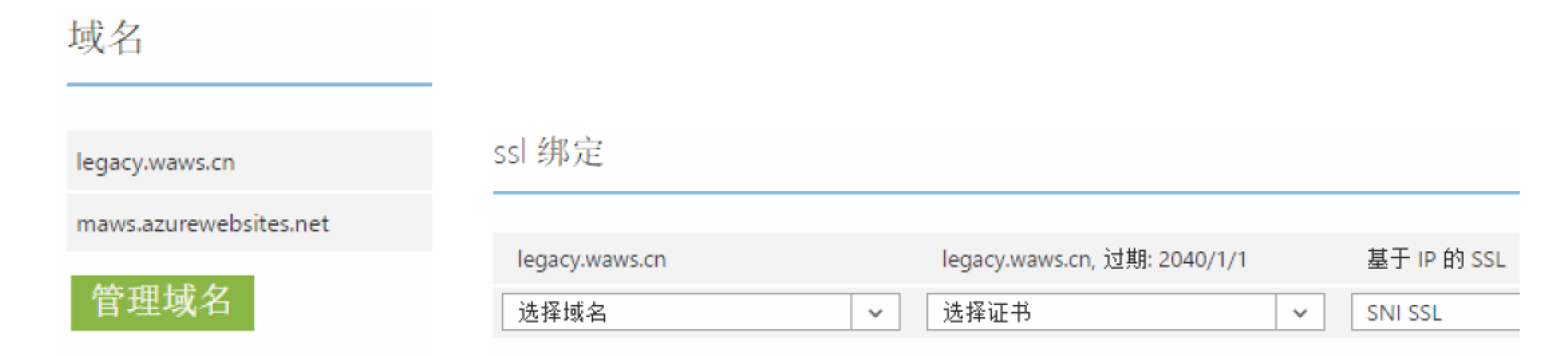


图 2-48　绑定 legacy.waws.cn 域名　　图 2-49　IP SSL 绑定

（3）打开浏览器，访问 https://legacy.waws.cn，此时显示服务器端采用的证书为 legacy.waws.cn。

（4）采用旧的浏览器，比如从 Windows XP 上访问网站时，服务器端采用的证书为 legacy.waws.cn，如图 2-50 所示。

2.6.4.2　配置 SNI SSL 绑定

配置完成 IP SSL 后，继续配置 modern.waws.cn，并采用 SNI SSL 绑定。

（1）绑定 modern.waws.cn 自有域名。

由于网站同时采用了 IP SSL 绑定，此时网站的 DNS 配置如图 2-51 所示。



图 2-50　legacy.waws.cn 证书

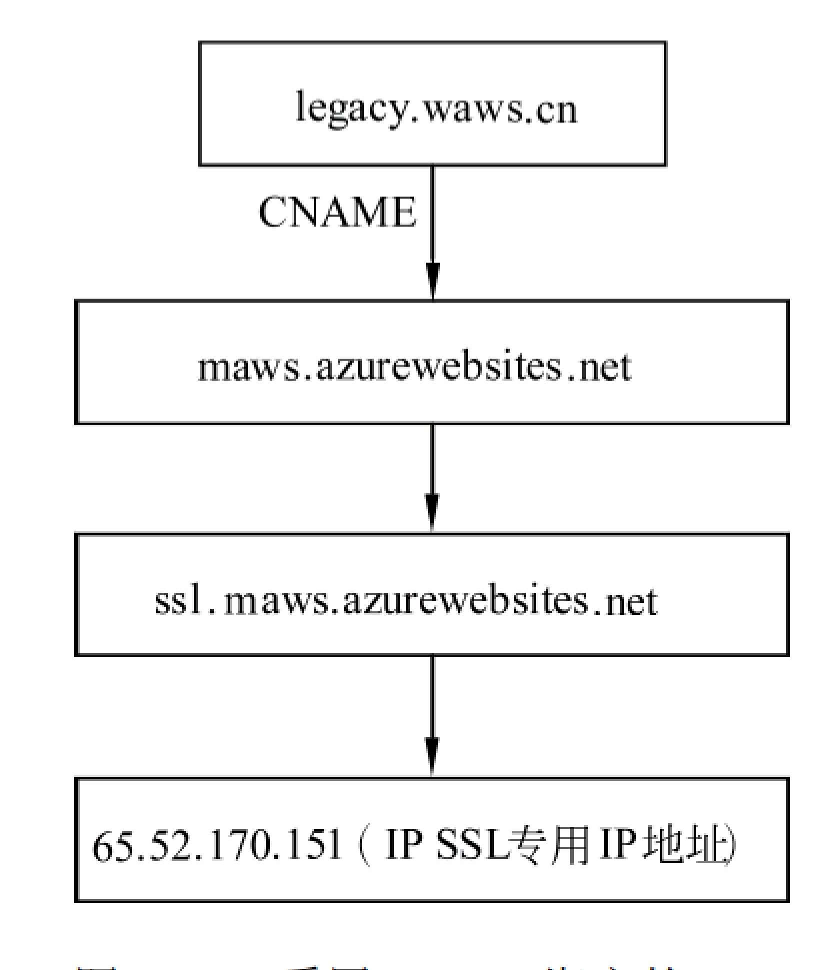


图 2-51　采用 IP SSL 绑定的 DNS

由于 maws.azurewebsites.net 已经指向专有 IP，如果采用 CNAME 配置，也将指向专有 IP 地址。前面介绍过 IP SSL 绑定只能绑定一个证书，因此会出现证书冲突的问题。

这种情况下，只能采用 A 记录的方式配置需要绑定 SNI SSL 的域名。如图 2-52 所示，需要配置 A 记录将 modern.waws.cn 指向部署单元的 IP 地址。

前面介绍过如何获取部署单元的 IP 地址。如图 2-53 所示，在管理域名的对话框中可以找到部署单元的 IP 地址。



图 2-52　配置 SNI SSL 域名的 A 记录　　　　图 2-53　获取部署单元 IP 地址

由于采用了 A 记录，因此同时需要配置 awverify 记录，所以需要配置如表 2-14 所示的两条 DNS 记录。

表 2-14　modern.waws.cn 域名配置

域　　名	DNS 记录类型	指　　向
Modern.waws.cn	A 记录	65.52.169.174
Awverify.modern.waws.cn	CNAME	awverify.maws.azurewebsites.net

（2）绑定 modern.waws.cn 域名。如图 2-54 所示，此时网站已经绑定了两个自有域名。

（3）创建 SNI SSL 绑定。如图 2-55 所示，在 Azure 门户管理网站创建一个 SNI SSL 绑定，并绑定到 modern.waws.cn 域名。

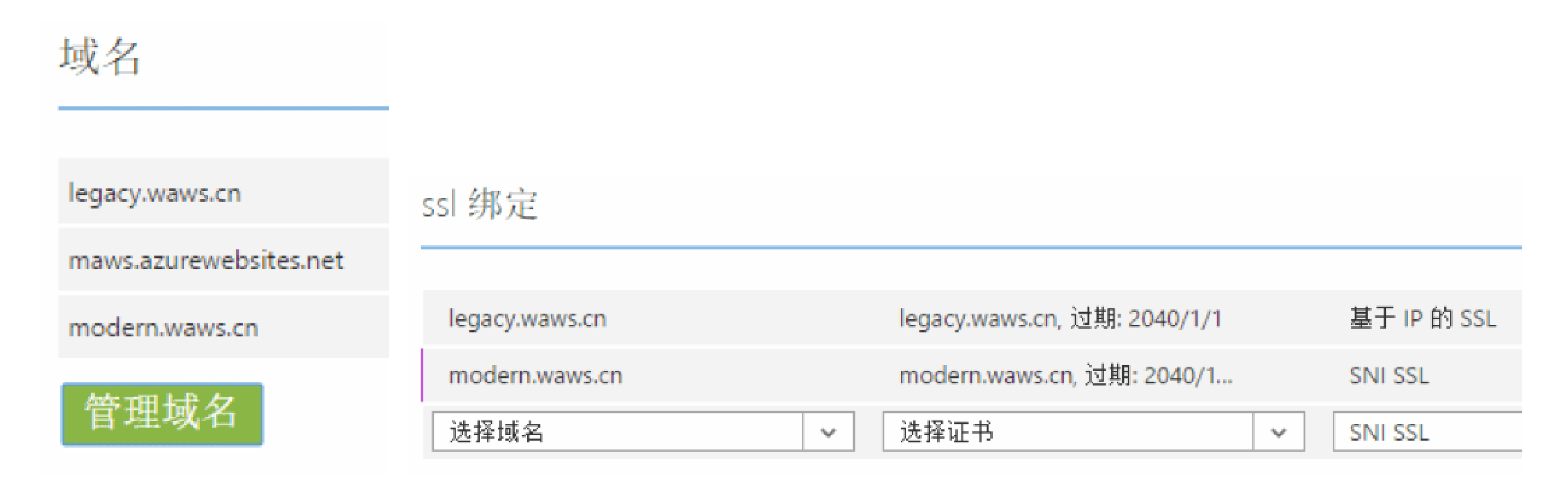


图 2-54　绑定 modern.waws.cn 域名　　　　图 2-55　创建 SNI SSL 绑定

（4）打开 IE 浏览器，访问 modern.waws.cn，采用的是 modern.waws.cn 证书，如图 2-56 所示。

2.6.4.3　配置清单

上面演示了如何同时配置 IP SSL 和 SNI SSL 的具体过程。要配置 Azure 网站支持同时采用 IP SSL 和 SNI SSL，需要以下资源和配置。

1. 资源

需要以下资源：



图 2-56　modern.waws.cn 证书

（1）两个自有域名。

（2）两个主题为自有域名的证书；或者一个通配符证书，比如*.waws.cn；或者一个 SAN 证书包含两个自有域名。

2．DNS 配置

同时采用 IP SSL 和 SNI SSL 的时候，绑定 IP SSL 的域名可以配置为 CNAME 或者 A 记录。但是绑定 SNI SSL 的域名只能采用 A 记录。

表 2-15 描述了绑定 IP SSL 的域名采用 CNAME 时的 DNS 配置列表。

表 2-15　DNS 配置清单

用　　途	域　　名	记录类型	指　　向
IP SSL	legacy.waws.cn	CNAME	maws.azurewebsites.net
SNI SSL	modern.waws.cn	A 记录	65.52.169.174（部署单元 VIP 地址）
鉴权	awverify.modern.waws.cn	CNAME	awverify.maws.azurewebsites.net

表 2-16 描述了绑定 IP SSL 的域名和绑定 SNI SSL 的域名同时采用 A 记录时的 DNS 配置列表。

表 2-16　DNS 配置清单

用　　途	域　　名	记录类型	指　　向
IP SSL	legacy.waws.cn	A 记录	65.52.170.151
鉴权	awverify.legacy.waws.cn	CNAME	awverify.maws.azurewebsites.net
SNI SSL	modern.waws.cn	A 记录	65.52.169.174（部署单元 VIP 地址）
鉴权	awverify.modern.waws.cn	CNAME	awverify.maws.azurewebsites.net

3．小结

（1）绑定 IP SSL 的域名，指向分配给 IP SSL 的专有域名。可以采用 A 记录或者 CNANE。

（2）绑定 SNI SSL 的域名，指向部署单元的 IP 地址，只能采用 A 记录。

（3）每个标准模式的网站可以使用一条 IP SSL 绑定和多条 SNI SSL 绑定。

2.6.5 强制客户使用 HTTPS 访问

Azure 支持通过 HTTP/HTTPS 的方式访问网站，比如 HTTP://www.contoso.com 或者 HTTPS: //www.contoso.com。但是很多商业网站要求客户必须使用 HTTPS 访问网站。下面两种方式强制客户使用 HTTPS 的方式访问 Azure 网站。

2.6.5.1 禁止 HTTP 访问

IIS 提供了相应的配置选项，可以禁止客户通过 HTTP 方式访问网站。在本地 IIS 上，可以很容易地通过 IIS 管理器或者 appcmd.exe 命令行工具来修改相应的配置。具体信息请参考下面的文档：

http://technet.microsoft.com/en-us/library/cc732367(v=WS.10).aspx

对于 Azure 网站，无法通过命令行或者管理工具来修改配置。可以将下面的配置内容合并到 Azure 网站的 web.config 中来禁止 HTTP 访问。

```
<system.webServer>
  <security>
    <access sslFlags="Ssl"/>
  </security>
</system.webServer>
```

此时，如果客户通过 HTTP://www.MyAzureSite.Net 来访问网站，客户会看到下面的错误信息：

```
您没有权限查看此目录或网页
```

很多客户看到这个错误信息后可能会非常迷惑或者紧张，他们可能会试图联系网站的客户服务来解决该问题。下面介绍如何使用 URL Rewrite 来更好地解决该问题。

2.6.5.2 使用 URL Rewrite

URL Rewrite 是 IIS 的一个扩展组件。它允许管理员通过定义规则来创建客户友好的 URL 或者搜索引擎优化的 URL。可以参考下面的文档来详细了解 URL Rewrite 的具体功能：

http://www.iis.net/learn/extensions/url-rewrite-module/using-url-rewrite-module-20

Microsoft Azure 网站的工作服务器上已经安装了 URL Rewrite，所以 Azure 网站也可以利用 URL Rewrite。下面的配置定义了一个 Rewrite 的规则，该规则自动将所有的 HTTP 请求重定向到 HTTPS 请求。

```
<rewrite>
  <rules>
```

```
      <rule name="Redirect to https">
        <match url="(.*)"/>
        <conditions>
          <add input="{HTTPS}" pattern="Off"/>
        </conditions>
        <action type="Redirect" url="https://{HTTP HOST}/{R:1}"/>
      </rule>
    </rules>
  </rewrite>
```

2.6.6　常见证书问题

客户在配置 Azure 网站 SSL 时可能会遇到很多问题。除了前面具体讨论过的 DNS 配置问题，还有很多问题与证书相关。

（1）提供的 Azure 网站证书必须具有私有密钥，通常是由密码保护的.PFX 格式的证书。在现代加密体系中，网站使用私有密钥来加密数据，而客户端使用公有密钥来解密数据。网站上使用的证书必须是具有私有密钥的证书。

（2）通常，需要向商业证书颁发机构（CA）申请和购买证书。在 Windows 操作系统中，默认安装了多个业界信任的证书颁发机构的根证书。可以在 Azure 网站上使用这些证书颁发机构颁发的证书。关于可信任的证书颁发机构，请参考下面的文档：

http://social.technet.microsoft.com/wiki/contents/articles/14215.windows-and-windows-phone-8-ssl-root-certificate-program-member-cas.aspx

如果网站的证书是企业自有 CA 服务器颁发的证书（比如由企业内部安装的 Windows CA 服务颁发的证书），称之为私有证书。这种证书不能用在 Azure 网站上，因为它们不被信任。如果不想通过被信任的证书颁发机构购买一个证书，那么如前面所示，可以选择使用自签名证书。通常，推荐在测试网站使用自签名证书。

（3）中间证书。基于安全考虑，所有的证书颁发机构已经不再直接颁发证书，而是通过中间证书颁发机构来颁发证书。中间证书颁发机构是根证书颁发机构下的从属颁发机构。但是，Microsoft Azure 网站不允许用户上传和安装中间证书。因此，在 Azure 网站上面使用的证书必须同时包含中间证书。如图 2-57 所示，在导出证书时，需要将中间证书一起导出。

（4）OSCP（Online Certificate Status Protocol，在线证书状态协议）与 CRL（Certificate Revocation List，证书注销列表）是维护证书状态的两种模式。OCSP 是比较新的方法，它克服了 CRL 的主要缺陷：必须经常在客户端下载以确保列表的更新。如果证书提供商提供的证书只支持 OSCP，那么证书应用在 Azure 网站上会有兼容性的问题。当网站的客户访问网站的时候，可能会收到无法获取证书有效状态的警告。要解决该问题，需要证书提供商重新提供一个同时包含 OSCP 和 CRL 信息的新证书。

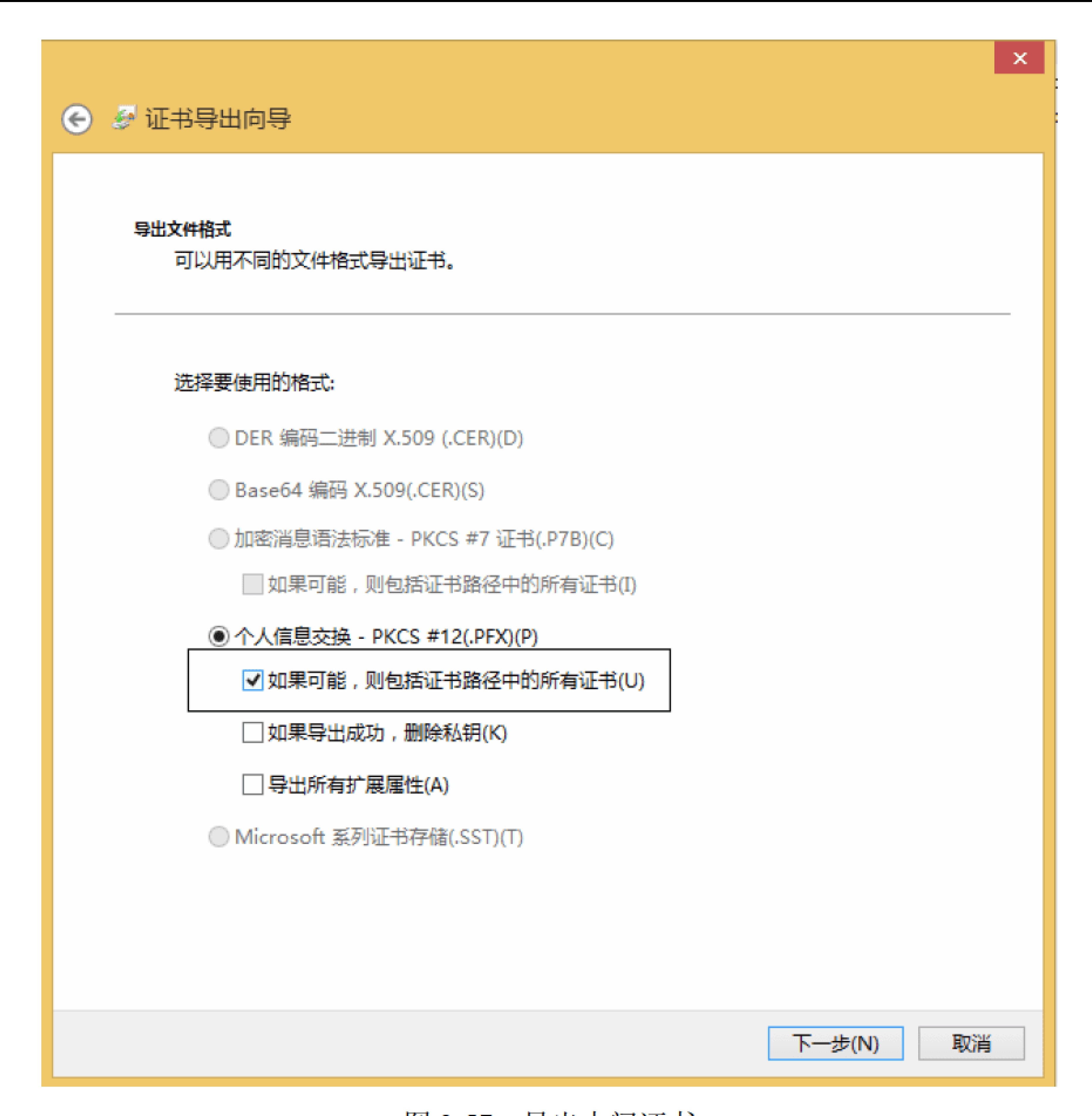


图 2-57　导出中间证书

2.7　监视网站

Microsoft Azure 网站通过管理门户网站提供了监视网站的功能。下面详细介绍如何监视网站。

2.7.1　仪表盘

在 Azure 管理门户网站的“仪表板”管理页面可以看到网站各项资源使用情况的图表，包括以下内容：

- CPU 时间。
- HTTP 请求数。
- 网络输出数据量。
- 网络输入数据量。
- HTTP 客户端错误。
- HTTP 服务器错误。

2.7.2　监视器

在 Microsoft Azure 管理门户网站的“管理”页面中，单击“监视器”选项卡以显示“监视器”管理页面。默认情况下，“监视器”页面上的图表显示的性能指标与“仪表板”页面上的图表显示的性能指标相同，如图 2-58 所示。

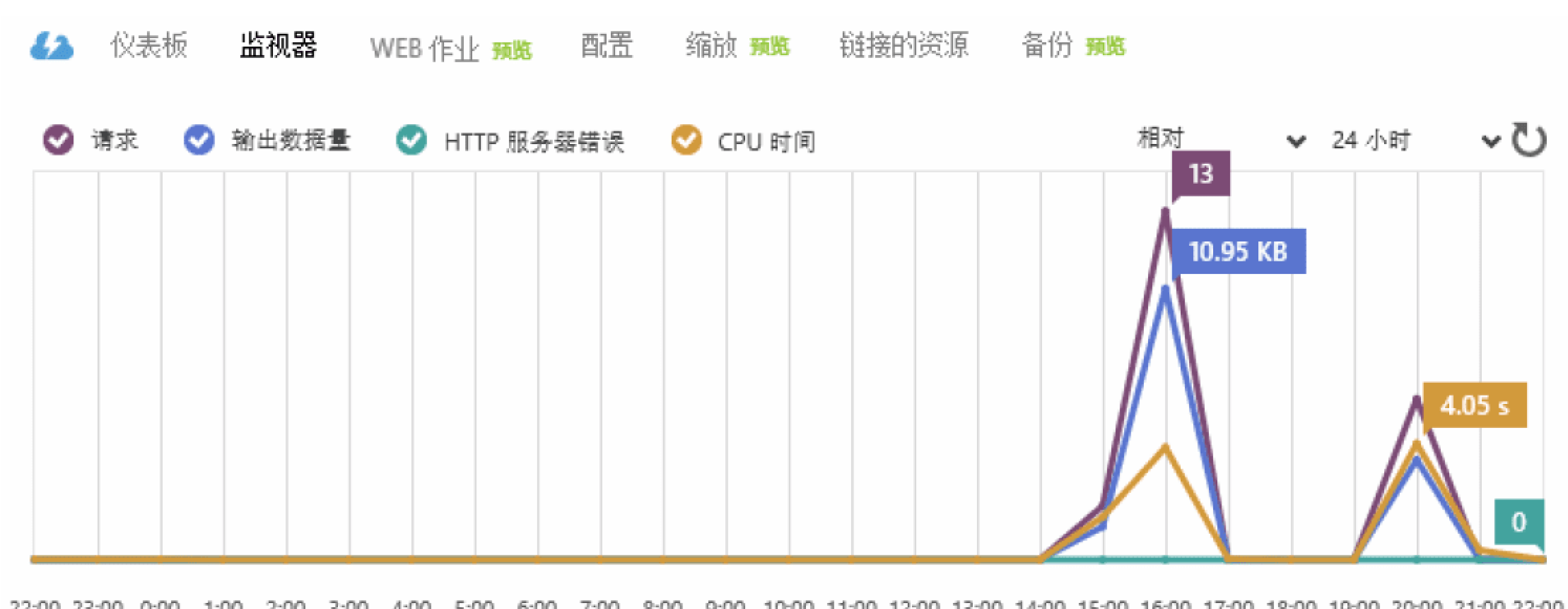


图 2-58　监视器页面

在监视器页面，可以通过单击页面底部的“添加度量值”来添加新的性能指标。

下面的列表描述了可以在“监视”页上的图表中查看的性能指标：

- CPU 时间：网站的 CPU 使用率。
- 请求数：客户端向网站发出的请求计数。
- 网络输出数据量：从网站已发送到客户端的数据量。
- 网络输入数据量：对网站从客户端处接收的数据的度量。
- HTTP 客户端错误：响应为 HTTP“4xx 客户端错误”的数目。
- HTTP 服务器错误：响应为 HTTP“5xx 服务器错误”的数目。
- HTTP 成功：响应为 HTTP“2xx 成功”的数目。
- HTTP 重定向：响应为 HTTP“3xx 重定向”数目。
- HTTP 401 错误：响应为 HTTP“401 未授权”数目。
- HTTP 403 错误：响应为 HTTP“403 已禁止”数目。

- HTTP 404 错误：响应为 HTTP“404 未找到”数目。
- HTTP 406 错误：响应为 HTTP“406 不可接受”的数目。

2.7.2.1　添加监视网络端点

在标准模式下，Microsoft Azure 网站允许配置网络端点来监视网站性能。最多可以配置两个网络监视端点，每个端点最多允许从 3 个地理位置来进行监控。

网络端点从用户配置的地理分布位置测试指定的 URL，并检测响应时间和 HTTP 状态代码。该测试从指定的地理位置每 5min 执行一次 HTTP GET 操作。如果响应时间大于 30s 或者 HTTP 状态代码大于等于 400，则认为网站出现故障，监控失败。

配置端点监控后，可以看到各个端点详细的响应时间和运行时间状态。

配置端点监测的步骤如下：

（1）登录到管理门户网站，打开“网站”选项卡，选中要配置的网站名称。

（2）单击“配置”选项卡。

（3）在“监视”部分输入端点的设置。

① 输入端点的名称。

② 输入要监控的 URL，比如 http://<mysite>.azurewebsites.com/monitor.html。该 URL 必须是网站域名下的 URL。

③ 从列表中选择一个或多个地理位置。

（4）可以重复前面的步骤来创建第二个端点。

（5）单击“保存”。

图 2-59 显示了网络监控端点的配置示例。该网站配置有两个监控端点，其中 EU 监控端点配置了两个测试地理位置，分别位于爱尔兰的都柏林和荷兰的阿姆斯特丹。每 5min，Microsoft Azure 会从这两个位置发送 HTTP GET 请求到 http://drumboy.azurewebsites.net/health.aspx 来监控网站的响应。如果响应时间超过 30s，或者 HTTP 状态码大于等于 400，则认为网站出现故障。

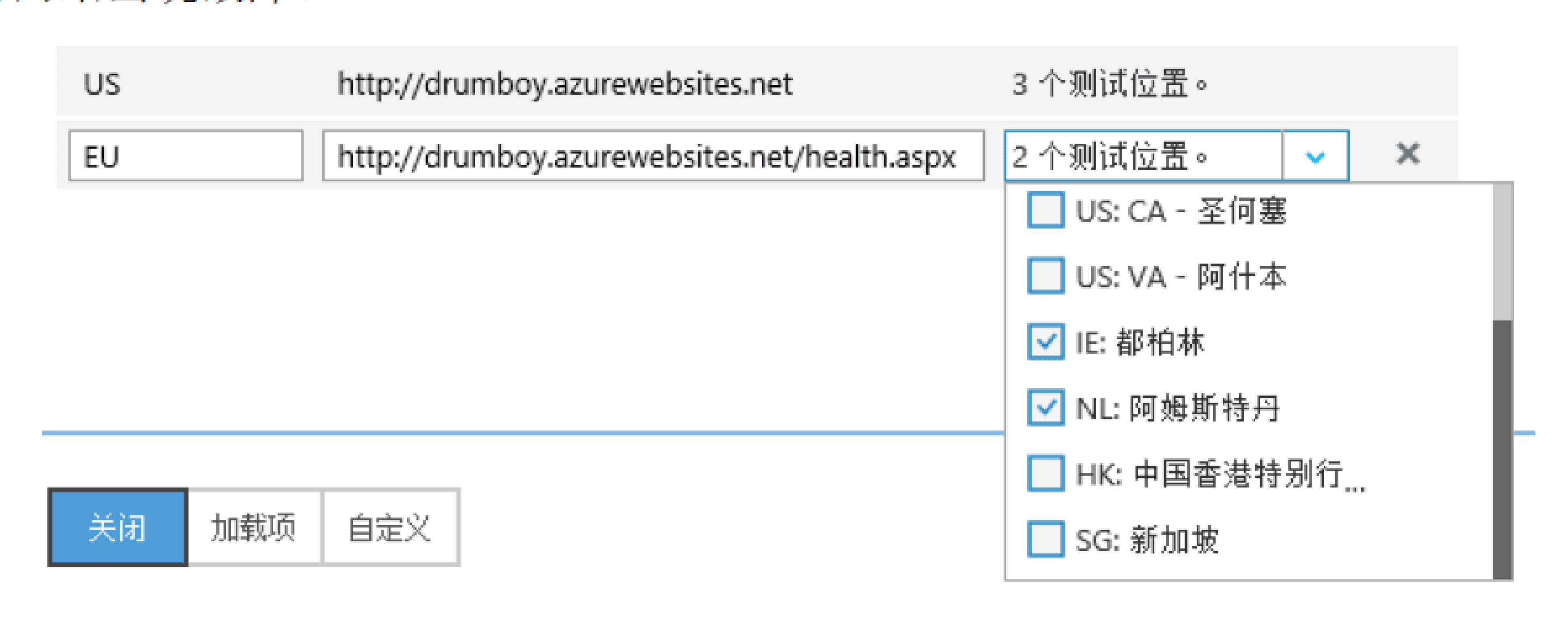


图 2-59　添加测试位置

添加网络端点监控后，可以在“监视”选项卡中观察每个地理位置的检测结果。如图 2-60 所示，选择“添加度量值”打开“选择度量值”窗口，单击“端点”，选择希望添加的检测项。



图 2-60　选择度量值

如图 2-61 所示，在监控面板单击“度量详细信息”，可以看到监控的具体情况。



图 2-61　查看测试结果

2.7.2.2　配置警报

可以基于网站性能指标或者监控端点的检测情况来定制警报规则。当警报被触发时，

Microsoft Azure 会向网站管理员发送邮件，及时地提示网站可能运行不正常。

配置警报规则的步骤如下：

（1）在“监控”面板，选择要设置警报的性能计数器。如果不在列表中，单击“添加度量值”来添加。

（2）单击“添加规则”，弹出“创建警报规则”对话框。

（3）如图 2-62 所示，指定警报名称和描述，单击“下一步”。



图 2-62 定义警报

（4）如图 2-63 所示，定义通知条件。如果在过去的 15min 内，都柏林的网络端点监控发现网站评价请求响应时间大于 30s 时，Microsoft Azure 自动给网站管理员发送警告邮件。



图 2-63 定义通知条件

（5）当警报被触发后，会收到包含下面内容的邮件：

'Response Time (Europe West/EU: IE- Dublin) Greater Than 30 (seconds) in the last 15 minutes' was activated for drumboy

2.8　扩 展 网 站

Azure 网站可以轻松扩展以满足用户持续增长的业务需求。Microsoft Azure 网站支持两种方式的扩展：纵向扩展和横向扩展。

纵向扩展，又称为向上扩展，通常是指通过提高机器性能来满足业务需要。横向扩展，又称为向外扩展，是指通过增加更多的机器实例来满足业务的需求。

横向扩展需要同时运行多台机器，通常消耗更多的能源。应用可能需要特殊的设计来满足横向扩展，而纵向扩展通常更容易实施。但是，横向扩展具有更灵活的硬件选择和更低的成本，硬件升级也更加容易。横向扩展将负载分布到几台机器上，在提供负载均衡功能的同时提供更高的容错能力。随着网站负载的增长，可以随时添加新的机器来满足业务要求。当负载下降时，可以释放多余的资以节约开支。与横向扩展不同，纵向扩展可以提高单台机器的吞吐量。图 2-64 描述了两种系统扩展方式的不同。

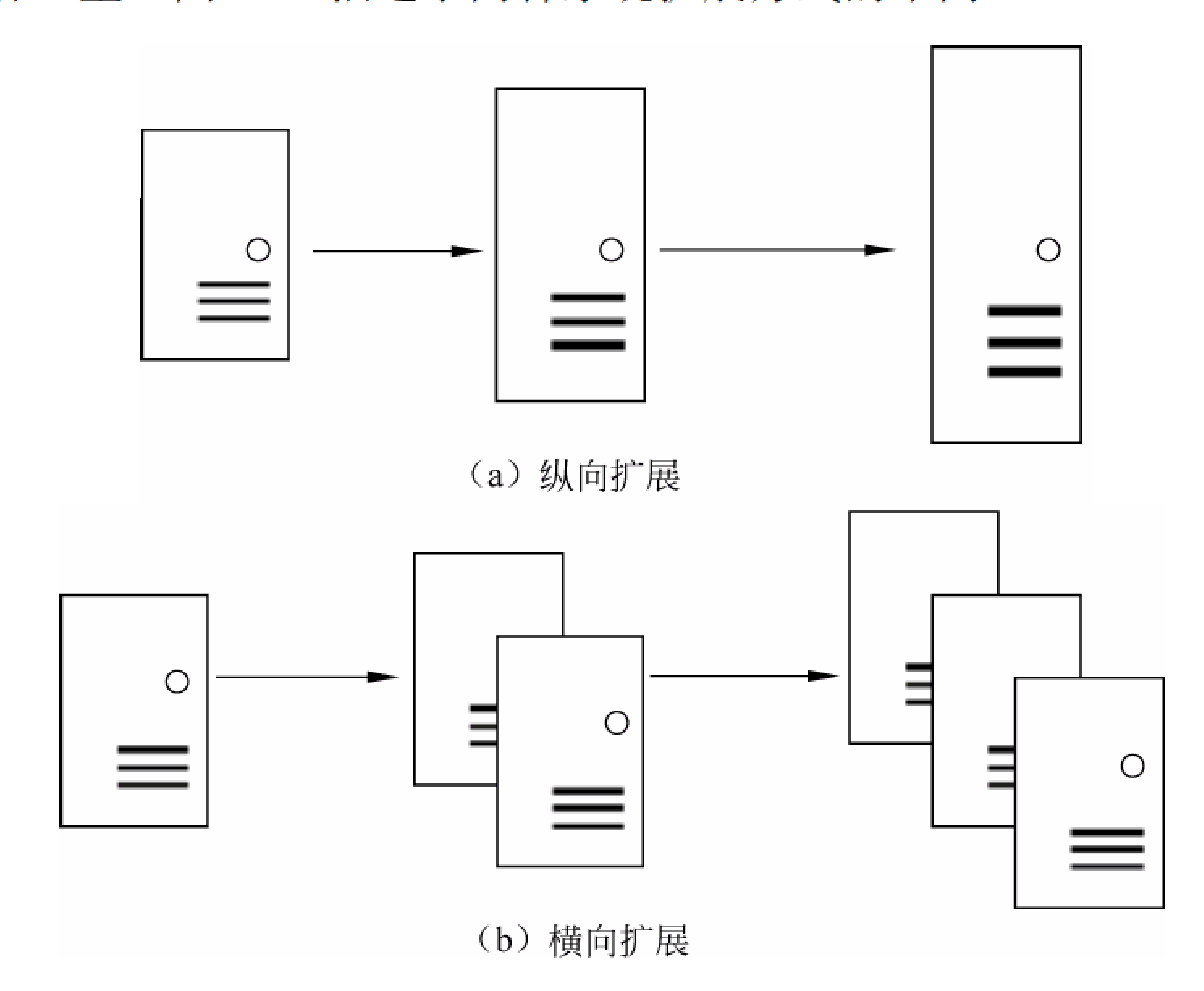


图 2-64　两种不同的系统扩展方式

2.8.1　如何选择扩展模式

通常，基于下面几点来选择扩展模式：

（1）持续可用性/冗余。必须假设故障是不可避免的，不管是硬件故障还是软件缺陷导致的故障。横向扩展的系统在很多情况下可以避免导致单点故障，降低服务停机时间。

（2）成本/性能灵活性。硬件成本和性能往往随着时间的推移迅速变化，在任何时间用

户都希望能够灵活地选择最佳配置，以优化成本/性能。如果用户的系统基于纵向扩展设计，那么用户就几乎锁定到一个满足条件的最低价格的硬件。但是横向扩展可以给用户带来更加灵活的硬件选择。

（3）持续升级能力。对一个单一的、大规模的系统进行升级或者添加新功能会更加困难，尤其是在不影响系统运行的情况下进行升级将是一个非常有挑战性的任务。将一套大的应用分解为独立可维护的服务，分布在不同的硬件上，可能是一个更好的选择。

（4）地理分布。跨国企业通常需要将关键应用部署到多个数据中心作为灾难冗余备份的解决方案。将应用部署到多个数据中心的另一个好处是可以更加接近客户，降低网络延迟。

2.8.2 如何扩展 Azure 网站

Azure 网站同时支持横向扩展和纵向扩展，为应用扩展提供了灵活多样的低成本扩展方式。

2.8.2.1 纵向扩展

Azure 提供了 4 个层次的站点模式：免费、共享、基本和标准，这 4 种模式对应不同的性能和功能，适用于不同的场景和应用。

在免费模式下运行的网站提供非常有限的资源配额和性能，建议采用免费模式站点进行开发任务或概念验证（proof of concept）工作。共享模式网站是一个低成本的可扩展模式，可提供高可用性和相比免费模式更好的性能。免费模式和共享模式不提供 SLA 的承诺。

在基本模式和标准模式下，网站独享一台或几台虚拟机。相比免费模式和共享模式，基本模式和标准模式网站将提供高可用性和更稳定的性能。同时，基本模式和标准模式提供了 3 种规格的虚拟机（小型、中型和大型）。表 2-17 列出了三种规格机器配置。

表 2-17 虚拟机规格

机器规格	CPU 内核数	内存/GB
小型	1	1.75
中型	2	3.5
大型	4	7

当业务需要纵向扩展时，有两个选择：

（1）升级网站模式。可以从共享模式升级到基本模式或标准模式。在基本模式和标准模式下，网站没有 CPU 使用率的限制。

（2）升级基本模式或标准模式下虚拟机的配置。

在基本模式和标准模式下，如果小型虚拟机不能满足需求，可以升级到中型或者大型虚拟机。

2.8.2.2 横向扩展

在共享模式下，网站最多可以有 6 个实例同时运行，即网站运行在 6 台虚拟机上，由

Azure 网站自动提供负载均衡。在基本模式下，网站最多可以有 3 个实例。在标准模式下，网站最多可以有 10 个实例同时运行。在业务繁忙时，可以增加机器实例以保证服务质量；在低谷时段，可以减少机器实例以节省成本。

2.8.2.3　自动横向扩展

在标准模式下，Microsoft Azure 网站支持自动横向扩展，可以根据业务需求定义自动横向扩展规则。当规则满足时，Microsoft Azure 网站自动扩展或者减少虚拟机数量。在业务高峰时能够无缝地及时扩容；在业务低谷时，自动释放资源以节省开支。

1．根据 CPU 使用率自动扩展

如果网站突发请求比较多，为了保证突发请求时的服务质量，可以考虑根据 CPU 使用率扩展。比如，可以设定在当前的机器 CPU 使用率达到 80%后，自动增加一台虚拟机。Microsoft Azure 网站自动为多台机器提供负载均衡。

可以按照下面的步骤设置网站根据 CPU 使用率自动扩展。

（1）登录到 Microsoft Azure 门户管理网站，选择要扩展的网站，单击“缩放”选项卡。

（2）在“容量”配置项下，可以看到“按指标缩放”选项。

（3）单击 CPU，如图 2-65 所示，可以配置实例数和目标 CPU 使用率。图 2-65 中配置最小实例数为 2，最大实例数为 4。目标 CPU 使用率为 60%～80%。它表示当网站启动时就有两台机器同时运行作为负载均衡。当两台机器的 CPU 使用率超过 80%时，自动增加第三台机器。当 3 台机器的 CPU 使用率超过 80%时，自动增加第四台机器。最多保持 4 台机器同时运行。当 CPU 使用率低于 60%时，自动减少一台机器。但是最少保持两台机器同时运行。为了避免频繁地增加/减少机器，CPU 使用率的计算周期为 5min。

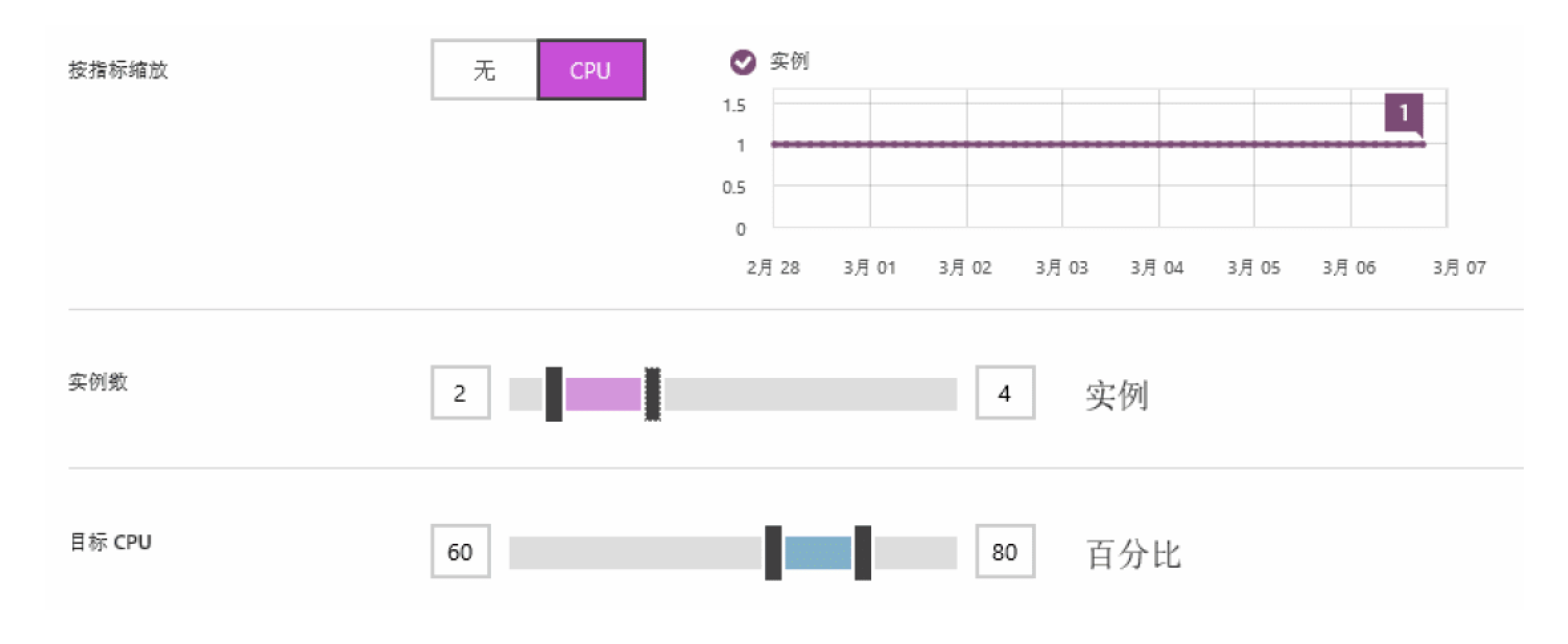


图 2-65　根据 CPU 使用率自动扩展

2．根据计划日程自动扩展

如果网站负载变化比较规律，比如在某个固定的时间客户访问比较频繁，可以考虑使用基于计划日程的自动扩展功能。可以按照下面的步骤设置网站根据日程自动扩展。

（1）登录到 Microsoft Azure 门户管理网站，选择要扩展的网站，单击“缩放”选项卡。

（2）在“容量”配置项下，可以看到“针对计划日程编辑缩放设置”选项。

（3）单击“设置计划时间”，如图 2-66 所示，可以看到“设置计划日程时间”对话框。



图 2-66 根据计划日程自动扩展

在对话框中，可以设置以下各项：

- 日夜的缩放设置不同。
- 工作日和周末的缩放设置不同，周末默认为周五晚 8 点到周一早 8 点。
- 设置每天的开始和结束时间，默认为早上 8 点至晚上 8 点。
- 设置时区，默认为用户的本地时区。
- 指定特定日期，比如图 2-66 指定了两个特定的日期：“双 11”和圣诞-元旦双节。最多可以指定 10 个特定日期。

（4）设置后，单击“保存”退出。

如表 2-18 所示，最多可以有 7 种定期计划日程（Y 表示选择，N 表示没有选择）。

表 2-18 自动扩展计划日程

日 程	日夜的缩放设置不同	工作日和周末的缩放设置不同	描 述
白天 夜间	Y	N	适用于网站白天和夜间访问量区别较大的情况。可以分别指定白天和夜间的实例数。比如，白天两个实例，夜间一个实例
工作日白天 工作日夜间 周末	Y	Y	适用于网站工作日白天和夜间访问量区别较大，同时工作日和周末访问量区别较大的情况。可以分别指定工作日夜间、工作日白天和周末的实例数。比如，工作日白天 1 台，工作日夜间 2 台，周末 2 台

续表

日　　程	日夜的缩放设置不同	工作日和周末的缩放设置不同	描　　述
工作日 周末	N	Y	适用于网站工作日和周末访问量区别较大的情况。可以分别指定工作日和周末的实例数。比如，工作日 1 台，周末 3 台

2.9　参考文献与扩展阅读

1. Server Name Indication

关于 Server Name Indication 的维基百科页面。该文章详细介绍了 SNI 背景知识、基本概念以及工作原理。

http://en.wikipedia.org/wiki/Server_Name_Indication

2. IIS SNI 功能

微软公司从 IIS 8 开始支持 SNI 功能。该文档详细介绍了在 IIS 中如何使用 SNI 功能。

http://www.iis.net/learn/get-started/whats-new-in-iis-8/iis-80-server-name-indication-sni-ssl-scalability

3. IIS CCS（Central Certificate Store）功能

在 Azure 网站中，SSL 的实现利用了 IIS 的 CCS（Central Certificate Store）功能。该文档详细介绍了如何在 IIS 中使用 CCS。

http://www.iis.net/learn/get-started/whats-new-in-iis-8/iis-80-centralized-ssl-certificate-support-ssl-scalability-and-manageability

4. Domain Name System

关于域名管理系统的维基百科页面。了解基本的 DNS 概念有助于在 Azure 网站上配置自有域名。该文章详细介绍了 DNS 的概念、实现和协议等，可以帮助读者了解 DNS 工作原理。

http://en.wikipedia.org/wiki/Domain_Name_System

5. Websites — Develop and deploy enterprise-grade Web apps

Azure 网站的官方文档，包含了开发、部署、配置和管理等文档。

http://azure.microsoft.com/en-us/documentation/services/websites/

6. Manage an Azure website

Azure 网站官方文档，详细介绍了如何管理 Azure 网站。

http://azure.microsoft.com/en-us/documentation/articles/web-sites-manage-azure-website/

7. Manage websites through the Azure Management Portal

Azure 网站官方文档，详细介绍了如何通过 Azure 管理门户网站管理用户的 Azure 网站。

http://azure.microsoft.com/en-us/documentation/articles/web-sites-manage/

8. Websites Pricing Details

Azure 网站官方文档，介绍了 Azure 网站的价格。

http://azure.microsoft.com/en-us/pricing/details/websites/

第 3 章　管理自动化

Microsoft Azure 提供了强大的管理门户网站，可以通过管理门户网站来管理 Azure 服务、网站和存储等。但是，很多情况下，用户更希望能够通过自动化来提高效率。例如：

- 人类厌恶重复的工作。如果需要通过管理门户网站创建 100 个网站，这是一件令人筋疲力尽的工作。如果还需要为每个网站指定对应的自定义域名，上传证书，修改配置，这将是令人崩溃的一件差事。
- 通过管理门户网站操作效率低，速度慢。同上，通过管理门户网站进行大批量和重复性的操作的时候不仅费力，而且费时间。
- 人类会犯错。当人类进行大批量、重复性的操作时很容易犯错。
- 机器/现有软件系统不能指挥人类。现有的 IT 系统不能通过指挥人类来集成并管理部署在 Azure 上的资源。

Microsoft Azure 提供了服务管理 REST API，可以通过编写程序调用 REST API 来管理 Azure 订阅和部署在 Azure 中的资源。

同时，基于 Azure 服务管理 REST API，微软公司提供了丰富的管理自动化方法，包括 Azure PowerShell 管理包、跨平台命令行管理程序和 Microsoft Azure 管理库（WAML）。如图 3-1 所示，管理门户网站、Azure PowerShell 管理包、跨平台命令行管理程序以及客户基于 WAML 开发的应用都是通过调用 Azure 管理 REST API 来管理 Azure 平台的资源。

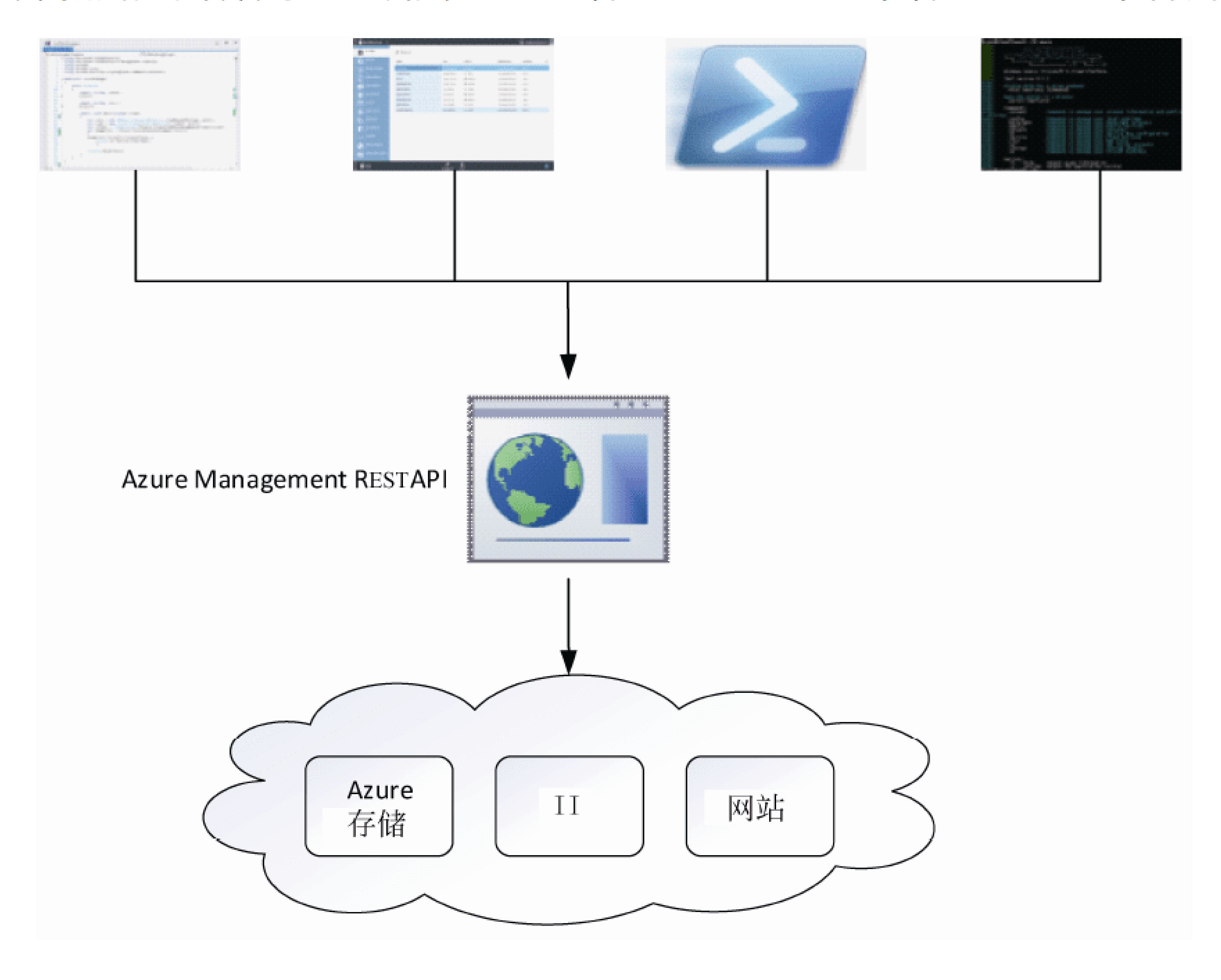


图 3-1　Azure 管理自动化

Windows PowerShell 是一个命令行外壳程序和脚本环境，使命令行用户和脚本编写者可以利用.NET Framework 的强大功能。Azure PowerShell 管理包提供了用于管理 Azure 资源的命令。利用这些命令，可以编写 PowerShell 脚本来管理部署在 Azure 中的资源。

跨平台命令行管理程序提供了用于管理 Azure 的各种命令。Azure PowerShell 只能运行在 Windows 平台上，而跨平台命令行管理程序可以运行在各种操作系统上。

Microsoft Azure 管理库则是基于 REST API 的一个.NET 封装库。如果用户希望开发自己的应用来管理 Azure，Azure 管理库可以使应用的复杂度大为降低，提高开发效率。

Azure Powershell、Azure 命令行和 Azure 管理库这 3 种方式并不支持 REST API 提供的所有功能。表 3-1 描述了上述三者对 2014 年 2 月的版本的 REST API 所支持的功能。在随后的版本中，将陆续加入其他功能的支持。

表 3-1 Azure 管理自动化支持的功能列表

服 务	PowerShell	命 令 行	Azure 管理库
Virtual Machine（虚拟机）	√	√	√
Cloud Service（云服务）	√		√
WebSites（网站）	√	√	√
Network（网络）	√	√	√
Storage（存储）	√	√	√
SQL Database（数据库）	√	√	√
Service Bus（服务总线）	√	√	√
Store（商店）	√		√
Scheduler（计划程序）			√
Monitoring（监视）			√
Traffic Manager（流量管理器）			√
HDInsight（大数据）	√	√	
Mobile Services（移动服务）		√	
Media Services（媒体服务）	√		√
WAAD Auth（活动目录认证）	√		√

3.1 Azure 环境

目前 Azure 有两个运行环境：AzureCloud（全球 Azure）和 AzureChinaCloud（中国区 Azure），默认环境为 AzureCloud。

这两个环境虽然采用完全相同的技术，但是完全相互独立。中国区 Azure 由微软公司和 21 世纪互联公司共同合作，由 21 世纪互联公司负责运营。目前，中国区环境有北京（China-North）和上海（China-East）两个数据中心。

此外，所有的自动化方式支持 Microsoft Azure Pack 环境。Microsoft Azure Pack 是微软公司推出的基于 Azure 技术的私有云产品。关于 Microsoft Azure Pack，可以访问下面的网

页获取更多信息：

http://www.microsoft.com/en-us/server-cloud/products/windows-azure-pack/default.aspx

关于两个环境的具体区别，请参考下面的文档：

http://msdn.microsoft.com/en-us/library/azure/dn578439.aspx

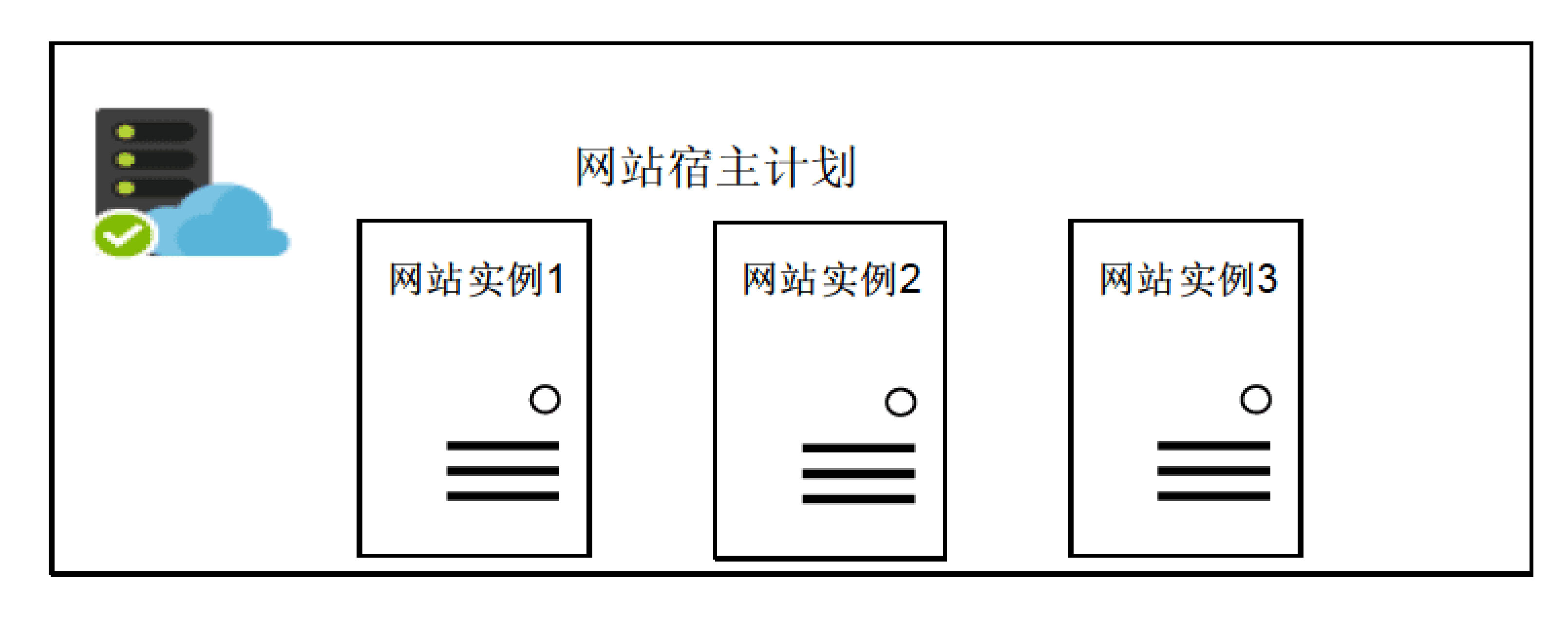
3.2　管 理 模 式

Azure PowerShell 和命令行提供了两种模式：

（1）服务管理模式。用于管理部署在 Azure 中的服务，比如云服务、虚拟机、Azure 网站、SQL Azure 等。服务管理模式是默认模式。

（2）资源管理模式。前面讲过，Azure 中所有的服务都隶属于某个资源组。比如，图 3-2 所示的资源组包含一个数据库和网站宿主计划，在网站宿主计划中同时包含 3 个网站实例。资源管理模式用于管理您的资源组。





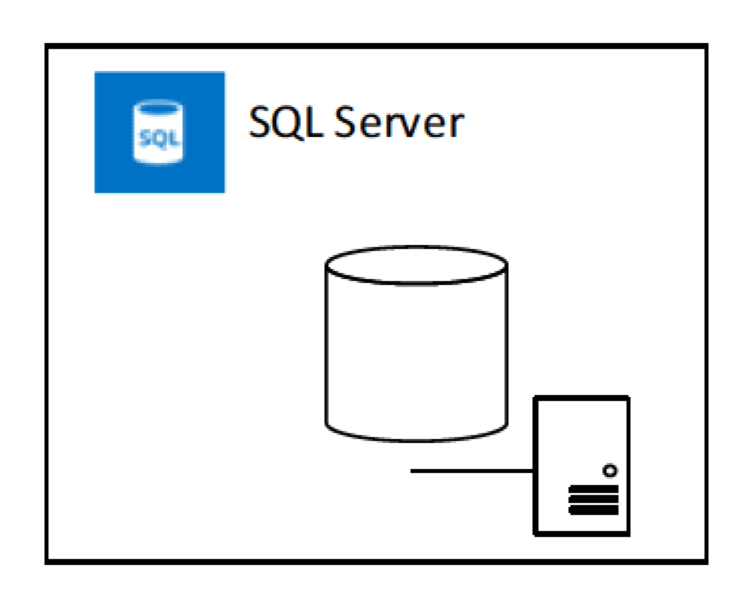


图 3-2　Azure 资源组概念

3.3　Azure 服务管理 API 客户端认证

Microsoft Azure 服务管理 API 不允许匿名调用。它支持两种认证方式：

（1）Azure 活动目录。

（2）管理证书。

3.3.1 使用 Azure 账户认证

Azure 账户是基于令牌的认证方式。它支持两种类型的账户：

（1）微软账户。即 Azure 订阅账户，比如<user>@outlook.com、<user>@hotmail.com 或者<user>@live.com。可以使用 Azure 订阅账户登录或者使用任何协同管理员账户登录。

（2）组织账户/Azure 活动目录。组织账户不同于微软账户，组织账户来源于 Azure 活动目录，比如 admin@contoso.onmicrosoft.com 或者 admin@contoso.com。

每一个 Azure 订阅都有一个默认的活动目录（default directory）。在默认的活动目录中，Azure 订阅账户已经添加为全局管理员。

3.3.2 通过管理证书认证

Microsoft Azure 管理证书用来验证客户端的 X.509 v3 证书，它与 Azure 订阅绑定。每个 Azure 订阅可以最多拥有 100 个证书。如果有多个订阅，并希望使用相同的管理证书来管理这些订阅，该证书必须与每个订阅相关联。

在 Azure 管理门户网站上绑定的证书是公钥证书，不包含私有密钥。在客户端，Microsoft Azure 管理证书必须是私钥证书。客户端证书必须有至少 2048 位的密钥长度，通常保存在个人证书库中。

使用管理证书时，需要注意以下两点：

（1）服务管理 API 不验证证书是否仍然有效。即使是一个过期的证书也可能验证成功。

（2）如果有多个管理证书，所有管理证书具有相同的权限。没有“基于角色”的认证。

可以通过两种方式配置管理证书：下载发布设置文件和上传管理证书。

3.3.2.1 下载发布配置文件

发布配置文件（publish settings file）是包含有关订阅信息的 XML 文件。它包含所有与 Live ID 相关联的订阅信息。同时，它包含可用于认证 Microsoft Azure 服务管理 API 的一个管理证书。可以通过下面的 URL 下载发布配置文件：

https://windows.azure.com/download/ publishprofile.aspx

默认文件名称为<Subscription_Name>-<Download_Date>-credentials.publishsettings。

通常情况下，发布配置文件形式如下：

```
<?xml version="1.0" encoding="utf-8"?>
<PublishData>
  <PublishProfile
    PublishMethod="AzureServiceManagementAPI"
    Url="https://management.core.windows.net/"
    ManagementCertificate="MIIKPAIBAz…QLk=">
```

```
    <Subscription
      Id="xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx"
      Name="My Subscription" />
  </PublishProfile>
</PublishData>
```

其中 ManagementCertificate 是用于管理 Azure 的管理证书。当下载发布配置文件时，Azure 自动创建一个新的管理证书，并把该证书与订阅关联。所以，可以使用包含在发布配置文件中的证书管理 Azure 订阅。每次下载发布配置文件， Azure 都会创建一个新的管理证书。

可以直接在代码中使用管理证书，比如，下面的代码根据 ManagementCertificate 的内容生成一个证书：

```
X509Certificate2 azureCert= new X509Certificate2(
    Convert.FromBase64String("Content of the ManagementCertificate");
```

或者可以使用工具根据发布配置文件生成证书，并将其导入个人证书库中。如果使用 Microsoft Azure PowerShell 或者 X-CLI 工具，这些工具会根据发布配置文件中的 ManagementCertificate 内容生成证书，并导入个人证书库中。

在 www.waws.cn 网站，可以找到本书作者发布的小工具。该工具可以将发布配置文件中的证书部分提取出来，并安装到本地证书库中。

3.3.2.2　上传自己的管理证书

同时，也可以上传自己的管理证书到 Microsoft Azure。与前面相同，以自签名证书为例加以介绍。

1．创建证书

可以通过 makecert 命令来创建自签名证书用于 Azure 管理 API 的认证。要创建自签名证书，以管理员身份运行 Visual Studio 命令行工具，然后运行以下命令：

makecert -sky exchange -r -n "CN=myAzureManagementCert" -pe -a sha1 -len 2048 -ss My " myAzureManagementCert.cer"

该命令产生两个证书。一个是带有私有密钥的证书，该证书自动保存到当前用户的个人证书库。同时，该命令产生一个.cer 文件，该文件是一个仅仅包含公有密钥的证书。需要上传.cer 文件用于认证。

2．上传证书

要上传管理证书到 Microsoft Azure，执行以下步骤：

（1）登录到 Microsoft Azure 管理门户。

（2）在左边的导航栏下方单击“设置”。

（3）在设置页面顶部单击“管理证书”选项卡。

（4）单击底部命令栏中的“上传”按钮，选择前面生成的.cer 文件。证书上传成功后，

会看到如图 3-3 所示的界面。指纹是证书各种属性的 SHA-1 哈希值。不同的证书具有不同的指纹，因此通常使用指纹来唯一标识一个证书。



图 3-3　管理证书列表

3.3.3　选择合适的认证方式

Azure 管理门户网站和客户端的工具（比如 Azure PowerShell、跨平台命令行和 Visual Studio）都支持基于 Azure 账户的认证。基于账户的认证是 Azure 最近才支持的方式。在此之前，客户只能选择基于管理证书的认证方式。

尽管 Azure 继续支持基于管理证书的认证方式，但是相对于基于账户的认证方式，基于管理证书的方式更加复杂，容易出错。如果可能，应该选择基于账户的认证方式。

基于账户的认证方法可以更容易地管理和访问 Azure 订阅。但是，这种方法产生的凭证只能使用 12 小时，过期后需要重新登录。对于需要长时间运行的应用来讲，这可能会中断管理自动化应用。对于这种情况，基于管理证书的认证方式是更好的选择。

3.4　使用 PowerShell 管理 Azure 网站

Microsoft Azure PowerShell 管理包是一个 PowerShell 模块，它是一个开源项目。可以用它来控制和自动化管理工作。通过 Windows PowerShell 管理包，可以创建管理、配置、监视部署在 Microsoft Azure 中的服务。可以在 PowerShell 命令控制台中以互动方式运行命令，或者编写 PowerShell 脚本完成复杂的管理任务。

本节详细介绍如何使用 Microsoft Azure PowerShell 管理 Azure 网站。

3.4.1　安装与运行

通过运行微软公司 Web 平台安装程序下载并安装 Microsoft Azure PowerShell 模块是最方便的方法。Web 平台安装程序可以通过下面的地址下载安装：

http://www.microsoft.com/web/downloads/platform.aspx

下载 Web 平台安装程序后，运行该程序，在搜索框中输入 Microsoft Azure PowerShell，在搜索结果中选择 Microsoft Azure PowerShell，然后单击“添加”按钮和“安装”按钮，如图 3-4 所示。

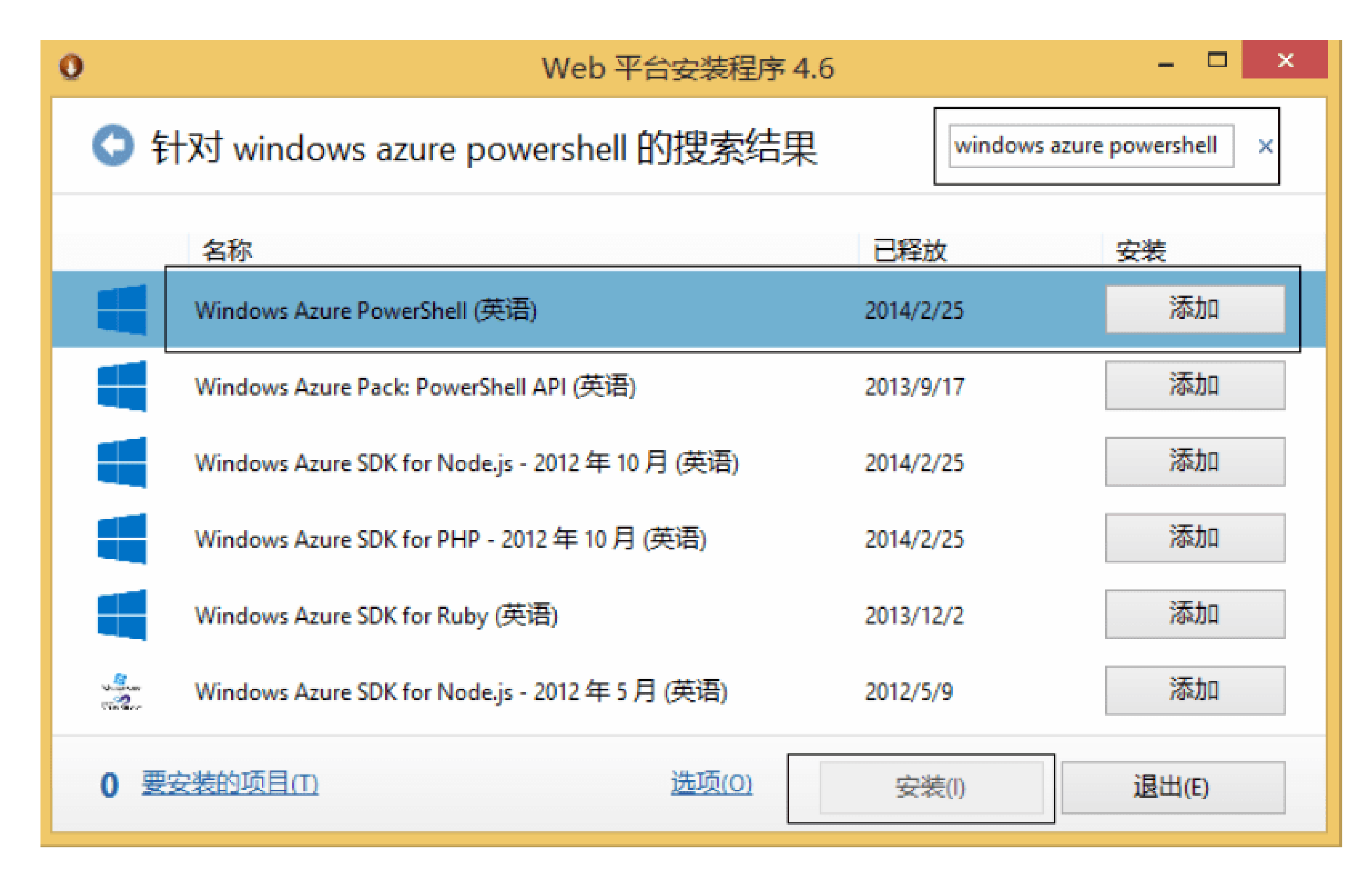


图 3-4　安装 Windows Azure PowerShell

Web 平台安装程序会自动安装 Microsoft Azure PowerShell 及其依赖的模块，比如 Microsoft Azure SDK，请按照提示完成安装。

Microsoft Azure PowerShell 模块还包括一个自定义的控制台。可以从标准的 Windows PowerShell 控制台或 Microsoft Azure PowerShell 控制台运行 Microsoft Azure PowerShell cmdlet。

在 Windows 8 或 Windows Server 2012 上，如图 3-5 所示，可以使用内置的搜索。从开始屏幕中选择“搜索”，输入 powershell。在搜索到的应用程序列表中包括 Windows PowerShell 和 Windows Azure PowerShell。可以单击其中任一应用程序来打开 Azure PowerShell 控制台窗口。

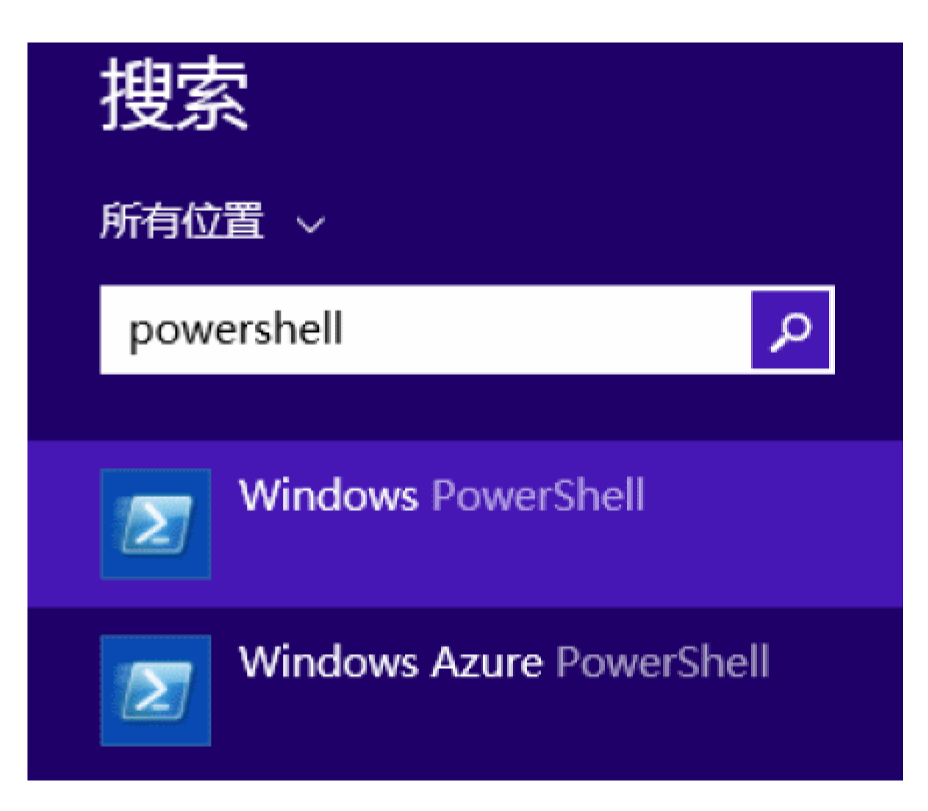


图 3-5　在 Windows 8 上打开 Azure PowerShell 控制台

3.4.2　查看 Azure 环境配置

可以运行 Get-AzureEnvironment 来查看 Azure 环境。默认环境为 AzureCloud。

```
PS C:\> Get-AzureEnvironment | Format-List -Property EnvironmentName
EnvironmentName : AzureCloud
```

```
EnvironmentName : AzureChinaCloud
```

获取中国 Azure 环境的具体信息：

```
PS C:\> Get-AzureEnvironment -Name AzureChinaCloud
Name                            : AzureChinaCloud
PublishSettingsFileUrl          : http://go.microsoft.com/fwlink/?LinkID=301776
ServiceEndpoint                 : https://management.core.chinacloudapi.cn/
ResourceManagerEndpoint         :
ManagementPortalUrl             : http://go.microsoft.com/fwlink/?LinkId=301902
ActiveDirectoryEndpoint         : https://login.chinacloudapi.cn/
ActiveDirectoryCommonTenantId    : common
ActiveDirectoryServiceEndpointResourceId    :https://management.core.
                                                China cloudapi.cn/
StorageEndpointSuffix           : core.chinacloudapi.cn
StorageBlobEndpointFormat       : {0}://{1}.blob.core.chinacloudapi.cn/
StorageQueueEndpointForma       : {0}://{1}.queue.core.chinacloudapi.cn/
StorageTableEndpointFormat      : {0}://{1}.table.core.chinacloudapi.cn/
GalleryEndpoint                 :
```

3.4.3 认证并连接到 Azure 订阅

在使用 Microsoft Azure PowerShell 之前必须通过认证。Microsoft Azure PowerShell 同时支持基于账户和基于管理证书的两种认证方式。

3.4.3.1 使用基于账户的认证方式

（1）打开 Microsoft Azure Powershell 控制台，并运行 Add-AzureAccount 命令。

（2）此时，弹出 Microsoft Azure 登录对话框，输入邮件地址和密码。

（3）Microsoft Azure 验证信息，验证通过后，会在 Azure PowerShell 控制台看到如下信息：

```
PS C:\> Add-AzureAccount
详细信息: Account "xxxxx@live.com" has been added.
详细信息: Subscription "Visual Studio Ultimate with MSDN" is selected as the
default subscription.
详细信息: To view all the subscriptions, please use Get-AzureSubscription.
详细信息: To switch to a different subscription, please use Select-
AzureSubscription.
```

如果需要连接到 AzureChinaCloud 环境，则需要指定 AzureChinaCloud 环境：

```
Add-AzureAccount -Environment AzureChinaCloud
```

3.4.3.2 使用基于管理证书的认证方式

当使用基于管理证书的认证方式时，只要订阅和证书是有效的，就可以管理和访问

Azure 订阅。这种方法更适用于自动化长时间运行的任务。然而，这种方法使得它难以管理访问共享订阅，例如管理一个具有多个授权用户的账户。使用管理证书的具体步骤如下：

（1）打开 Microsoft Azure PowerShell 控制台。

（2）运行以下命令：

```
Get-AzurePublishSettingsFile
```

请注意，如果订阅基于 AzureChinaCloud，那么需要运行下面的命令：

```
Get-AzurePublishSettingsFile -Environment AzureChinaCloud
```

此命令会打开 IE 窗口，并导航到 Microsoft Azure 订阅文件下载页面，根据提示下载并保存发布配置文件（.publishsettings 文件）

（3）回到 Microsoft Azure PowerShell 控制台，执行下面的命令：

```
Import-AzurePublishSettingsFile  <your publishsettings file>
```

命令具体执行情况如下所示：

```
PS C:\> Get-AzurePublishSettingsFile
PS C:\> import-AzurePublishSettingsFile 'C:\wzhao.publishsettings'
详细信息: Setting: Visual Studio Ultimate with MSDN as the default and current
subscription. To view other subscriptions use Get-AzureSubscription
```

3.4.4　管理网站

3.4.4.1　Azure 网站相关命令

Microsoft Azure PowerShell 提供了很多管理网站的命令。在 Azure PowerShell 控制台中运行下面的命令，可以列出所有的 Azure 网站相关的命令：

```
PS C:\> get-command -module Azure | where-object {$ .name -match "website"}
| select name

Name
----
Disable-AzureWebsiteApplicationDiagnostic
Disable-AzureWebsiteDebug
Enable-AzureWebsiteApplicationDiagnostic
Enable-AzureWebsiteDebug
Get-AzureWebsite
Get-AzureWebsiteDeployment
Get-AzureWebsiteJob
Get-AzureWebsiteJobHistory
Get-AzureWebsiteLocation
Get-AzureWebsiteLog
```

```
New-AzureWebsite
New-AzureWebsiteJob
Publish-AzureWebsiteProject
Remove-AzureWebsite
Remove-AzureWebsiteJob
Restart-AzureWebsite
Restore-AzureWebsiteDeployment
Save-AzureWebsiteLog
Set-AzureWebsite
Show-AzureWebsite
Start-AzureWebsite
Start-AzureWebsiteJob
Stop-AzureWebsite
Stop-AzureWebsiteJob
Switch-AzureWebsiteSlot
Update-AzureWebsiteRepository
```

3.4.4.2 获取命令帮助

可以运行 Get-help 命令来获得每条 cmdlet 的具体帮助信息和示例。比如，下面的命令列出 Get-AzureWebsite 的详细帮助信息。

```
Get-help Get-AzureWebsite -full
```

3.4.4.3 列出用户的网站配置

Get-AzureWebsite 命令可以列出用户的网站，下面是一个示例输出：

```
C:\> get-azurewebsite
Name            : yuebing
State           : Running
Host Names      : {yuebing.azurewebsites.net}
```

如果希望获取某个特定网站的详细信息，可以运行下面的命令列出指定网站的相关详细信息：

```
PS C:\> get-azurewebsite -Name yuebing
NumberOfWorkers              : 1
DefaultDocuments             : {Default.htm, Default.html, Default.asp,
                                 index.htm...}
NetFrameworkVersion          : v4.0
PhpVersion                   : 5.4
RequestTracingEnabled        : False
HttpLoggingEnabled           : False
DetailedErrorLoggingEnabled : False
PublishingUsername           : $yuebing
PublishingPassword           : xxxxxxx
AppSettings                  : {WEBSITE_NODE_DEFAULT_VERSION}
```

```
Metadata                      : {}
ConnectionStrings             : {}
HandlerMappings               : {}
Name                          : yuebing
State                         : Running
HostNames                     : {yuebing.azurewebsites.net}
WebSpace                      : eastasiawebspace
SelfLink                     :https://waws-prod-hk1-001.api.azure websites.
                              windows.net:454/20130801/websystems/websites/
                              web/ subscriptions/166e9762-9cac-4a8e-8b81-5659
                              a1fe1181/webspaces/eastasiawebspace/ sites/yuebing
RepositorySiteName            : yuebing
Owner                         :
UsageState                    : Normal
Enabled                       : True
AdminEnabled                  : True
EnabledHostNames              : {yuebing.azurewebsites.net, yuebing.scm.
                                azurewebsites. net}
SiteProperties               :Microsoft.WindowsAzure.Commands.Utilities. Websites.
                              Services. WebEntities. SiteProperties
AvailabilityState             : Normal
SSLCertificates               : {}
SiteMode                      : Limited
HostNameSslStates             : {}
AzureDriveTraceEnabled        :
AzureDriveTraceLevel          : Error
AzureTableTraceEnabled        :
AzureTableTraceLevel          : Error
ManagedPipelineMode           : Integrated
WebSocketsEnabled             : False
RemoteDebuggingEnabled        : False
RemoteDebuggingVersion        : VS2012
```

3.4.4.4　创建一个新网站

如果想创建一个网站，可以运行 New-AzureWebsite 命令。比如下面的命令在东亚数据中心（香港）创建一个名为 hktestsitewzhao 的新网站：

```
New-AzureWebsite -Name hktestsitewzhao -Location "East Asia"
```

3.4.5　资源管理模式

Azure Powershell 资源管理模式提供了管理 Azure 资源和资源组的命令。首先需要运行下面的命令切换到 Azure 资源管理模式：

```
Switch-AzureMode AzureResourceManager
```

3.4.5.1 获取资源组信息

Get-AzureResourceGroup 命令返回 Azure 订阅下所有的资源组。默认返回所有资源，可以通过指定-name 参数选择特定的资源组。

下面是一个实际运行的例子，返回信息中包含了资源的名称、类型以及数据中心。

```
PS C:\> get-azureresourcegroup

ResourceGroupName    : Default-SQL-EastAsia
Location             : eastasia
ProvisioningState    : Succeeded
Resources            :
                Name         Type                              Location
               ==========   ===============================   ========
               p1yinaoor6   Microsoft.Sql/servers              eastasia
               drumboy       Microsoft.Sql/servers/databases    eastasia

ResourceGroupName    : Default-Web-WestUS
Location             : westus
ProvisioningState    : Succeeded
Resources            :
             Name             Type                       Location
             ============     ========================= ========
             Default1         Microsoft.Web/serverFarms  westus
             contoso7100      Microsoft.Web/sites        westus
             ContosoWAADSite Microsoft.Web/sites        westus

ResourceGroupName    : drumboy
Location             : eastasia
ProvisioningState    : Succeeded
Resources            :
          Name     Type                                     Location
          =======  ======================================== ========
          drumboy  Microsoft.ClassicCompute/domainNames      eastasia
          drumboy  Microsoft.ClassicCompute/virtualMachines eastasia
```

3.4.5.2 创建一个新的资源组

New-AzureResourceGroup 命令创建一个新的 Azure 资源组。下面的命令在东亚（香港）数据中心创建一个名为 contosoPowerShell 的资源组：

```
PS C:\> new-azureResourceGroup -name contosoPowerShell -Location "East Asia"
详细信息: 11:49:48 - Create resource group 'contosoPowerShell' in location
'East Asia'

ResourceGroupName        : contosoPowerShell
Location                 : eastasia
```

```
ProvisioningState         : Succeeded
Resources                 :
```

3.4.5.3　创建一个网站并加入指定的资源组

New-AzureResource 命令可以创建一个网站并将其加入指定的资源组。下面的命令创建个网站（ContosoWebByPS）并加入到资源组 contosoPowershell 中：

```
PS C:\> $WebsiteProperties = @{name = "ContosoWebByPS"; computeMode =
"Shared"; siteMode = "Limited";}

PS C:\> New-AzureResource -Name ContosoWebByPS -ResourceGroupName contoso
PowerShell -ResourceType 'Microsoft.Web/sites' -Location "East Asia"
-ApiVersion 2014-04-01 -PropertyObject $WebsiteProperties
详细信息: 11:53:08 - Resource group "contosoPowerShell" is found.
详细信息: 11:53:08 - Creating resource "ContosoWebByPS" started.
详细信息: 11:53:18 - Creating resource "ContosoWebByPS" complete.

Name                      : ContosoWebByPS
ResourceGroupName         : contosoPowerShell
ResourceType              : Microsoft.Web/sites
ParentResource            :
Location                  : East Asia
Properties                : {
                               "name": "ContosoWebByPS",
                               "state": "Running",
…
```

当创建网站并加入指定的资源组时，Azure 自动创建一个名为 DefaultX 的宿主计划，并将网站加入到该计划。X 是数字，从 0 开始。通过 Get-AzureResourceGroup 命令，可以查看网站和自动创建的宿主计划。宿主计划对应的资源类型为 Microsoft.Web/serverFarms。

```
PS C:\> get-azureResourcegroup -name contosoPowershell

ResourceGroupName         : contosoPowerShell
Location                  : eastasia
ProvisioningState         : Succeeded
Resources                 :
          Name                     Type                         Location
          ======================   ==========================   ========
          Default1                 Microsoft.Web/serverFarms    eastasia
          ContosoWebByPS           Microsoft.Web/sites          eastasia
```

3.4.5.4　创建一个网站宿主计划

管理门户网站没有提供单独创建网站宿主计划的功能。只能在创建网站的时候选择同时创建新的网站宿主计划。使用 Azure Powershell 可以单独创建一个宿主计划。下面的命令创建一个 Free 模式的宿主计划。

```
PS C:\> $WebHostPlanProperties = @{name = "ContosoWebHostPlanByPS"; sku
```

```
="Free";}
PS C:\> New-AzureResource -Name ContosoWebHostPlanByPS -ResourceGroupName
contoso PowerShell -ResourceType 'Microsoft.Web/serverFarms' -Location
"East Asia" -ApiVersion 2014-04-01 -PropertyObject $WebHostPlanProperties
详细信息: 12:03:24 - Resource group "contosoPowerShell" is found.
详细信息: 12:03:25 - Creating resource "ContosoWebHostPlanByPS" started.
详细信息: 12:03:28 - Creating resource "ContosoWebHostPlanByPS" complete.

Name                : ContosoWebHostPlanByPS
ResourceGroupName   : contosoPowerShell
ResourceType        : Microsoft.Web/serverfarms
ParentResource      :
Location            : East Asia
Properties          : {
                        "name": "ContosoWebHostPlanByPS",
                        "sku": "Free",
                        "workerSize": 0,
                        "numberOfWorkers": 0,
                        "currentWorkerSize": 0,
                        "currentNumberOfWorkers": 0,
                        "status": 0,
                        "webSpace": "contosoPowerShell-EastAsiawebspace"
                    }
```

此时，运行 Get-AzureResourceGroup 命令可以看到新创建的宿主计划：

```
PS C:\> get-azureResourcegroup -name contosoPowershell

ResourceGroupName   : contosoPowerShell
Location            : eastasia
ProvisioningState   : Succeeded
Resources           :
                      Name                   Type                        Location
                      ====================== =========================== ========
                      ContosoWebHostPlanByPS Microsoft.Web/serverFarms   eastasia
                      Default1               Microsoft.Web/serverFarms   eastasia
                      ContosoWebByPS         Microsoft.Web/sites         eastasia
```

3.4.5.5　通过模板创建资源组

前面演示了如何创建一个空的资源组，然后通过 New-AzureResource 命令创建新的资源加入资源组。Azure PowerShell 同时提供了更便捷的方式——利用资源组模板。资源组模板基于 JSON 格式，里面包含了一个复杂的 Azure 服务所需要的资源定义，比如名称、资源类型、资源数量等。

Azure 提供了一个资源组模板库，其中包含许多常见场景和其他用户上传的模板。可以通过 Get-AzureResourceGroupGalleryTemplate 命令查看现有的模板。或者使用 Save-Azure

Resource GalleryTemplate 命令将模板保存到本地。也可以自定义一个资源组模板，当使用自定义的资源组模板时，该模板自动被加入到模板库中。

下面演示如何通过模板创建一个包含网站和 SQL Azure 数据库的资源组。

（1）获取所有微软公司发布的模板：

```
PS C:\> Get-AzureResourceGroupGalleryTemplate -Publisher Microsoft

Publisher          Identity
---------          --------
Microsoft          Microsoft.ApacheTomcat7.0.1.0-preview
Microsoft          Microsoft.ASPNETEmptySite.0.1.0-preview
Microsoft          Microsoft.ASPNETEmptySite.0.2.0-preview
Microsoft          Microsoft.ASPNETStarterSite.0.1.0-preview
Microsoft          Microsoft.ASPNETStarterSite.0.2.0-preview
Microsoft          Microsoft.Bakery.0.1.0-preview
Microsoft          Microsoft.Bakery.0.2.0-preview
Microsoft          Microsoft.Boilerplate.0.1.0-preview
Microsoft          Microsoft.Boilerplate.0.2.0-preview
Microsoft          Microsoft.Cache.0.2.0-preview
Microsoft          Microsoft.ClassicStorage.0.2.0-preview
Microsoft          Microsoft.HTML5EmptySite.0.1.0-preview
Microsoft          Microsoft.HTML5EmptySite.0.2.0-preview
Microsoft          Microsoft.Jetty.0.1.0-preview
Microsoft          Microsoft.PersonalSite.0.1.0-preview
Microsoft          Microsoft.PersonalSite.0.2.0-preview
Microsoft          Microsoft.PhotoGallery.0.1.0-preview
Microsoft          Microsoft.PhotoGallery.0.2.0-preview
Microsoft          Microsoft.PHPEmptySite.0.1.0-preview
Microsoft          Microsoft.PHPEmptySite.0.2.0-preview
Microsoft          Microsoft.PHPStarterKit.0.1.0-preview
Microsoft          Microsoft.PHPStarterKit.0.2.0-preview
Microsoft          Microsoft.ServiceGatewayManagementConsole.0.1.0-preview
Microsoft          Microsoft.ServiceGatewayManagementConsole.0.2.0-preview
Microsoft          Microsoft.SQLDatabase.0.2.0-preview
Microsoft          Microsoft.TeamProject.0.2.0-preview
Microsoft          Microsoft.WebSite.0.1.0-preview1
Microsoft          Microsoft.WebSite.0.2.0-preview
Microsoft          Microsoft.WebSiteMySQLDatabase.0.2.0-preview
Microsoft          Microsoft.WebSiteSQLDatabase.0.2.0-preview
```

（2）获取模板详细信息。

接下来，利用 Microsoft.WebSiteSQLDatabase.0.2.0-preview 模板来创建包含网站和 SQL Azure 数据库的资源组。在创建资源之前，先看看模板的具体信息。

```
PS C:\> Get-AzureResourceGroupGalleryTemplate
```

```
                -Identity Microsoft.WebSiteSQLDatabase.0.2.0-preview

Identity                : Microsoft.WebSiteSQLDatabase.0.2.0-preview
Publisher               : Microsoft
Name                    : WebSiteSQLDatabase
Version                 : 0.2.0-preview
CategoryIds             : {azure, web, data, showInVS}
PublisherDisplayName: Microsoft
DisplayName             : Website + SQL
DefinitionTemplates : https://gallerystoreprodch.blob.core.windows.net/
                        prod-microsoft-windowsazure-gallery/8D6B920B-10
                        F4-4B5A-B3DA-9D398FBCF3EE.PUBLICGALLERYITEMS.
                        MICROSOFT.WEBSITESQLDATABASE.0.2.0-PREVIEW/
                        DeploymentTemplates/
                        Website NewHostingPlan SQL NewDB-Default.json
Summary                 : Enjoy secure and flexible development, deployment, and
                        scaling options for your web app plus a SQL database.
Description             : ……
```

（3）查看模板文件内容。

在 IE 中打开模板定义文件的 URL，可以查看 JSON 格式的模板文件内容。可以通过修改一个现有模板来创建自己的模板。

下面的命令将模板文件保存到本地：

```
PS C:\> Get-AzureResourceGroupGalleryTemplate
        -Identity Microsoft.WebSiteSQLDatabase.0.2.0-preview
        | Save-AzureResourceGroupGalleryTemplate -Path c:\AzureTemplates\

Path
----
C:\AzureTemplates\Microsoft.WebSiteSQLDatabase.0.2.0-preview.json
```

（4）获取对应的参数信息。

利用模板创建资源组的时候，通常需要指定参数，比如网站名称、数据库名称等。有 3 种指定参数的方式：

- 模板参数文件（JSON 格式）。
- 在运行命令时提供。
- 通过命令行指定参数。

下面的命令显示所需要的参数、参数的类型和默认值。有默认值的参数为可选参数。

```
PS C:\> $template=get-content -raw
        -path C:\AzureTemplates\Microsoft.WebSiteSQLDatabase.0.2.0-preview.json
       | ConvertFrom-Json
```

```
PS C:\> $template.parameters
siteName                 : @{type=string}
hostingPlanName          : @{type=string}
siteLocation             : @{type=string}
sku                      :@{type=string;allowedValues=System.Object[];default
                         Value=Free}
workerSize               :@{type=string;allowedValues=System.Object[];default
                         Value=0}
serverName               : @{type=string}
serverLocation           : @{type=string}
administratorLogin       : @{type=string}
administratorLoginPassword  : @{type=securestring}
databaseName             : @{type=string}
collation                :@{type=string; defaultValue=SQL_Latin1_ General_CP1_
                         CI_AS}
```

（5）利用模板创建资源组。

有两种方式用于指定模板：本地文件路径和 URL。Save-AzureResourceGalleryTemplate 可以将已有的模板保存到本地。下面演示利用保存到本地的模板文件创建资源组。

首先需要创建参数对象：

```
PS C:\> $params = @{siteName="ContosoSiteFromTemplate";
                   hostingPlanName="ContosolPlanFromTemplate";
                   siteLocation="East Asia";
                   serverName="contososqlsrv";
                   serverLocation="East Asia";
                   administratorLogin="contososqlsrv";
                   administratorLoginPassword="pwd";
                   databaseName="contososqldb"}
```

然后通过模板文件创建新的资源组：

```
PS C:\> New-AzureResourceGroup -Name ContosoRPFromTemplate -Location "East
Asia"
-TemplateUri "https://gallerystoreprodch.blob.core.windows.net/
            prod-microsoft-windowsazure-gallery/
            8D6B920B-10F4-4B5A-B3DA-9D398FBCF3EE.
            PUBLICGALLERYITEMS.MICROSOFT.WEBSITESQLDATABASE.0.2.0-PREVIEW/
            DeploymentTemplates/Website NewHostingPlan SQL NewDB-Default.json"
-TemplateParameterObject $params

详细信息： 22:52:26 - Create resource group 'ContosoRPFromTemplate' in
location 'East Asia'
详细信息: 22:52:27 - Template is valid.
…
```

3.4.5.6 获取更新

Azure PowerShell 是一个开源项目，可以在 GitHub 上面了解到该项目的最新进展。目前，Azure PowerShell 大概每个月都会有一个更新。另外，每当 Azure 产品发布新功能，Azure PowerShell 都会相应地推出新版本。

https://github.com/Azure/azure-sdk-tools

也可以通过 Web 平台安装程序来检查并安装更新。

3.5 使用跨平台命令行管理网站

Azure 跨平台命令行（Xplat-CLI）提供了用于管理 Azure 的一套开源的、跨平台的命令。Xplat-CLI 提供了许多与 Microsoft Azure 管理门户相同的功能，比如管理 Web 站点、虚拟机、移动服务、SQL 数据库以及 Microsoft Azure 平台提供的其他服务。

Xplat-CLI 是用 JavaScript 编写的，它基于 Microsoft Azure SDK for Node.js 实现，通过 Node.js 运行，因此可以运行在不同的平台上。与 Azure PowerShell 相同，Xplat-CLI 是基于 GitHub 的开源项目。

https://github.com/WindowsAzure/azure-sdk-tools-xplat

本节将详细介绍如何安装和配置 Microsoft Azure 跨平台命令行，以及如何使用它来管理 Microsoft Azure 网站。

3.5.1 安装

可以直接运行下面的安装程序来安装 Xplat-CLI。该安装程序同时会自动安装 Node.js。

http://go.microsoft.com/fwlink/?linkid=275464&clcid=0x409

如果已经安装了 Node.js，可以运行下面的命令来安装 Xplat-CLI：

npm install azure-cli

3.5.2 查看 Azure 环境

与 PowerShell 相同，默认环境为 AzureCloud。

```
C:\>azure account env list
info:    Executing command account env list
data:    Name
data:    ---------------
data:    AzureCloud
data:    AzureChinaCloud
info:    account env list command OK
```

3.5.3　连接到 Azure 订阅

3.5.3.1　使用基于管理证书的认证方式

（1）下载 Azure 订阅信息文件，在命令行控制台运行以下命令：

```
azure account download
```

如果下载基于 Azure 中国的订阅信息文件，在命令行控制台运行以下命令：

```
azure account download -e AzureChinaCloud（Azure 中国）
```

此命令会打开 IE 窗口，并导航到 Microsoft Azure 订阅文件下载页面。根据提示下载并保存发布配置文件（.publishsettings 文件）。

（2）回到命令行控制台，执行下面的命令连接到 Azure 订阅：

```
azure account import {path to .publishsettings file}
```

如果有多个 Microsoft Azure 的订阅，下载的.publishsettings 文件将包含所有订阅信息。使用 azure account import 命令导入.publishsettings 文件时，其中一个订阅将被选择为执行操作时所使用的默认订阅。可以使用 azure account list 命令查看订阅，以及哪一个是默认的订阅，此命令将返回类似于下面的信息：

```
Info:    Executing command account list
data:    Name           Id                                   Current
data:    -----------    --------------------------------     --------
data:    Azure-sub-1    ####################################    true
data:    Azure-sub-2    ####################################    false
```

在上面的列表中，Current 列表示当前默认的订阅是 Azure-sub-1。要更改默认订阅，请使用 azure account set 命令，例如下面的命令将默认订阅改为 Azure-sub-2：

```
azure account set Azure-sub-2
```

3.5.3.2　使用基于 Azure 账户的认证方式

使用基于账户的认证方式，只需要运行下面的命令：

```
azure login -u<userEmail>-p<password>
azure login -u<userEmail>-p<password>-e AzureChinaCloud（Azure 中国）
```

注意：在 Windows 平台上使用 Azure 活动目录登录时，认证令牌（token）会被缓存在本地用户的目录下，即 C:\Users\<You Account Name>\.azure\azureProfile.json。建议在结束管理任务后运行 Azure logout 命令退出当前登录。Azure logout 命令会清除缓存的令牌。

3.5.3.3　CLI 命令帮助

在命令行控制台，直接运行 azure 命令会返回 Xplat-CLI 的基本信息和支持的 Azure 服

务。如果想列出所有 Azure 网站相关的命令，请运行 azure site 命令。

大多数的命令格式如下所示：

```
azure <服务> <操作> 【参数】
```

比如：azure site list。

3.5.4 管理网站

与 PowerShell 相比，Xplat-CLI 支持的网站管理功能更丰富。下面介绍几个简单的命令。

3.5.4.1 创建网站

下面的命令在东亚数据中心（香港）创建一个名为 clitestsite 的网站：

```
azure site create --location "East Asia" clitestsite
```

3.5.4.2 列出网站

azure site list 命令列出用户的所有网站，下面是一个输出：

```
C:\windows\system32>azure site list
info:    Executing command site list
+ Getting locations
+ Getting sites
data: Name         Slot   Status   Location    Mode        URL
data: ------       ----   ------   --------   ------       -----------
data:  clitestsite        Running  East Asia  Standard     clitestsite. Azureweb
                                                           sites.net
info: site list  command OK
```

3.5.4.3 配置自有域名

可以运行 azure site domain 命令管理自有域名，下面的命令添加一个自有域名到指定的网站：

```
azure site domain add<YourCustomDomainName><YourAzureSiteName>
```

下面的命令将 www.contoso.com 域名与 clitestsite azurewebsites.net 绑定在一起：

```
azure site domain add www.contoso.com clitestsite。
```

3.5.4.4 管理证书

CLI 提供了管理证书的命令，目前 Azure PowerShell 还不支持管理网站证书。具体命令如下：

```
site cert list [options] [name]
site cert add [options] <certificate-path> [name]
site cert delete [options] <thumbprint> [name]
```

```
site cert show [options] <thumbprint> [name]
```

3.5.4.5　配置扩展

通过 azure site scale 命令可以配置网站的缩放/扩展选项，比如下面的命令将 clitestsite 升级/降级为共享模式网站：

```
C:\windows\system32>azure site scale mode free clitestsite
info:    Executing command site scale mode
+ Updating a site configuration
info:    site scale mode command OK
```

3.5.4.6　查看日志

同 PowerShell 一样，CLI 也提供了用于查看日志和诊断信息的命令。

（1）下载日志和诊断信息：

```
azure site log download<sitename>
```

（2）实时查看日志和诊断信息：

```
azure site log tail<sitename>
```

（3）修改日志和诊断配置：

```
azure site log set<sitename>
```

3.6　使用 REST API 管理网站

Microsoft Azure 服务管理 API 提供了通过编程管理 Microsoft Azure 服务的功能。Microsoft Azure 服务管理 API 是一套 REST API、XML 格式和 JSON 格式。所有的 API 调用都基于 SSL。其中，Microsoft Azure 网站管理 REST API 提供了核心的站点管理功能，包括：

（1）创建、删除和配置网站。

（2）查询网站的状态。

（3）查询性能指标，比如资源使用情况、配额和限制。

（4）获取发布配置文件等信息。

（5）网站备份与恢复。

3.6.1　Azure 网站管理员角色

Microsoft Azure 网站管理 REST API 的客户角色主要分为两大类：

（1）网站管理员。负责创建和管理 Microsoft Azure 网站及相关资源。这个角色对应 Microsoft Azure 订阅的管理员或协同管理员。

（2）发布者。发布者可以访问网站的后台文件和数据，并使用如 FTP 或 Web Deploy 协议来更新/发布网站的内容。从 Microsoft Azure 网站的角度来看，发布者是一个类似开发

人员的角色，发布者角色可以更新网站内容，但是不能修改网站的配置。

3.6.2 资源结构

Microsoft Azure 网站管理 REST API 提供的用于管理和部署网站的资源层次结构如下：

```
/subscriptions
   /webspaces
      /sites
            /config
            /publishxml
            /usages
            /metrics
            /repository
   /serverfarm
```

对应于 REST API 的 URL 如下：

```
HTTP://management.core.windows.net/<subscriptionid>/services/webspaces/
<webspace name>/sites/<site name>/
```

表 3-2 描述了上述资源的具体意义。

表 3-2 REST API 资源描述

资 源	描 述
subscriptions	Microsoft Azure 订阅
webspaces	一个网站空间是与用户订阅在一个数据中心的相关联的逻辑实体。用户在一个给定数据中心的所有网站都属于网站空间
sites	用户的 Microsoft Azure 网站
config	用户网站大的设置，比如 AppSettings、ConnectionStrings、错误日志和.NET Framework 版本等配置
publishxml	XML 格式的文件，它包含用户用于发布 Web 应用程序到 Microsoft Azure 网站的设置。该文件可以导入并从 Visual Studio 或 Web Matrix 使用
usages	包含有关当前网站的资源使用信息
metrics	历史资源使用信息
repository	与网站相关的源代码控制管理存储库，比如 Git
serverfarm	即 Web 宿主计划

3.6.2.1 支持的操作

1. 创建资源

新资源使用 HTTP POST 操作创建。如果资源创建成功，则返回一个 HTTP 201（创建）状态码。如果客户试图创建一个已经存在的资源，服务器端返回 HTTP 409（冲突）状态码。

2. 获取资源状态

读取已存在的资源的当前状态与配置等。客户端使用 HTTP GET 操作获取资源状态。

如果资源存在，则响应状态码为 HTTP 200。如果资源不存在，则响应的状态码是 HTTP 404（未找到）。

3．更新/修改资源

HTTP PUT 请求用于更新/修改现有的资源。如果资源更新成功，则返回一个 HTTP 200 状态码。如果要更新的资源不存在，则返回一个 HTTP 404（未找到）状态码。

4．删除资源

如果想删除一个现有的资源，需要通过 HTTP DELETE 请求。如果资源被成功删除，则返回一个 HTTP 200 状态码。如果对象已被删除或不存在，则返回 HTTP 404。

3.6.3　身份认证

Microsoft Azure 服务管理 REST API 不允许匿名调用。所有的请求都必须使用管理证书认证，并通过双向 SSL 以确保对服务提出的要求是安全的。Microsoft Azure 管理证书用来验证客户端的 X.509 v3 证书。Microsoft Azure 的管理证书上传到 Microsoft Azure 并与订阅绑定。每个 Microsoft Azure 的订阅可以最多拥有 100 个证书。如果有多个订阅，并希望使用相同的管理证书来管理这些订阅，该证书必须与每个订阅相关联。

在客户端，Microsoft Azure 管理证书必须有至少 2048 位的密钥长度，应保存在个人证书库中。客户端的证书应该包含证书的私钥。上传到 Microsoft Azure 管理门户的证书则必须是不含私钥的.cer 格式的文件。

使用管理证书时，需要注意以下两点：

（1）服务管理 API 不验证证书是否仍然有效。即使是一个过期的证书也可能验证成功。

（2）如果有多个管理证书，所有管理证书具有相同的权限。没有“基于角色”的认证。

3.6.4　应用实例

使用 Azure 网站管理 REST API 需要自己编写代码。在开始编写代码之前，需要具备以下条件：

- 一个有效的 Microsoft Azure 订阅。
- 安装 Visual Studio 2012 或者 2013。
- 安装 Windows Azure SDK（可以通过 Web Platform Installer 安装）。
- 已经上传有效的管理证书（具体步骤详见 3.3 节）。

下面简单介绍如何通过 Azure 网站管理 REST API 来管理 Azure 网站。在下面的例子中，列出所有的网站名称和网站模式。

（1）打开 Visual Studio，新建一个 Visual Studio 命令行工程，命名为 ManageWebSitesUsing RestAPI。

（2）安装 Json.Net 扩'展包。

单击“工具”，选择 Library Package Manager→Package Manager Console。在控制台中

运行 Install-Package NewtonSoft.Json。

```
PM> Install-Package NewtonSoft.Json
Installing 'Newtonsoft.Json 6.0.1'.
Successfully installed 'Newtonsoft.Json 6.0.1'.
Adding 'Newtonsoft.Json 6.0.1' to ManageWebSitesUsingRestAPI.
    Successfully added 'Newtonsoft.Json 6.0.1' to ManageWebSitesUsingRestAPI.
```

（3）打开 Program.cs 文件，在 Program 类之前加入下面的代码。这些代码定义了一个类描述站点的基本信息。

```
class webSiteBasics
{
    private string  name;
    private string  siteMode;
    public webSiteBasics(string name, string site mode)
    {
        this._name = name;
        this._siteMode = site_mode;
    }
    public string Name
    {
        get { return  name; }
        set { this. name = value; }
    }
    public string SiteMode
    {
        get { return  siteMode; }
        set { this. siteMode = value; }
    }
}
```

（4）在 Program.cs 文件中，在 main 函数前加入下面的代码。请用图 3-4 中的订阅 ID 和指纹代替 xxxxxx。

```
private const string ThumbPrint = "xxxxxx";
private const string Version = "2014-04-01";
private const string SubscriptionId = "xxxxxx";
```

为了给客户提供最好的产品和服务，Microsoft Azure 大约每三四个月发布一次更新。每次更新后，相应的管理 REST API 也会有更新。当调用管理 REST API 时，需要指定版本。为了保持兼容性，REST API 更新后，之前的版本不受影响。如果需要使用新功能，则需要指定新的版本号。如果没有指定版本或者指定了错误的版本，服务器端返回 400 Bad Request。关于版本的更多信息，请参考下面的文档：

http://msdn.microsoft.com/en-us/library/windowsazure/gg592580.aspx

（5）定义查找证书的函数，该函数在当前用户的证书库中根据指定的证书指纹来查找

证书。之前运行 makecert 命令生成两个证书，公有密钥已经上传到 Azure 管理门户网站。现在使用的是包含私有密钥的证书。

```
private static X509Certificate2 GetStoreCertificate(string thumbprint)
{
    foreach (var location in locations)
    {
        X509Store store = new X509Store("My", StoreLocation.CurrentUser);
        try
        {
            store.Open(OpenFlags.ReadOnly | OpenFlags.OpenExistingOnly);
            X509Certificate2Collection certificates = store.Certificates.
            Find(
                X509FindType.FindByThumbprint, thumbprint, false);

            if (certificates.Count >= 1)
            {
                return certificates[0];
            }
        }
        finally
        {
            store.Close();
        }
    }

    throw new ArgumentException(string.Format(
        "A Certificate with Thumbprint '{0}' could not be located.",
        thumbprint));
}
```

（6）编写函数，获取所有的 WebSpaces。要获取所有的 WebSpaces，发送下面的请求：

```
GET https://management.core.windows.net/{SubID}services/WebSpaces
```

下面的例子中指定返回的结果格式为 JSON，并使用了 JSON.NET 来处理结果。最后，函数返回一个字符串列表，该列表包含了所有 WebSpaces 的名称。

```
private  static  List<string>  ListWebSpaces(string  subscriptionId,  X509
Certificate2 certificate, string version)
{
    string uriFormat = https://management.core.windows.net/{0}/
     + "services/WebSpaces";
    Uri uri = new Uri(String.Format(uriFormat, subscriptionId));
    HttpWebRequest request = (HttpWebRequest)HttpWebRequest.Create(uri);
    request.Method = "GET";
```

```
        request.Headers.Add("x-ms-version", version);
        request.ClientCertificates.Add(certificate);
        request.ContentType = "application/json";
        HttpWebResponse response;
        response = (HttpWebResponse)request.GetResponse();
        StreamReader sr = new StreamReader(response.GetResponseStream());
        var result = sr.ReadToEnd();
        List<string> webSpaces = new List<string>();
        JArray jWebSpaces = (JArray)JsonConvert.DeserializeObject(result);
        foreach (var webSpace in jWebSpaces)
        {
            webSpaces.Add(webSpace["Name"].ToString());
        }
        return webSpaces;
    }
```

（7）编写函数，获取指定的 WebSpaces 下的所有站点信息。返回格式为 JSON 格式，使用 JSON.NET 来解析返回结果。最终，函数返回一个 webSiteBasics 类的列表。

```
    private static List<webSiteBasics> ListWebSites(string subscriptionId,
    X509Certificate2 certificate, string webSpace, string version)
    {
    string uriFormat = https://management.core.windows.net/{0}/
       + "services/WebSpaces/{1}/sites";
        Uri uri = new Uri(String.Format(uriFormat, subscriptionId, webSpace));
        HttpWebRequest request = (HttpWebRequest)HttpWebRequest.Create(uri);
        request.Method = "GET";
        request.Headers.Add("x-ms-version", Version);
        request.ClientCertificates.Add(certificate);
        request.ContentType = "application/json";
        HttpWebResponse response;
        response = (HttpWebResponse)request.GetResponse();
        StreamReader sr = new StreamReader(response.GetResponseStream());
        var result = sr.ReadToEnd();
        JArray jWebSites = (JArray)JsonConvert.DeserializeObject(result);
        List<webSiteBasics> webSites = new List<webSiteBasics>();
        foreach (var webSite in jWebSites)
        {
            webSiteBasics siteBasics = new webSiteBasics(website ["Name"].
            ToString(),
                webSite["SKU"].ToString() );
            webSites.Add(siteBasics);
        }

        return webSites;
    }
```

（8）修改 main 函数，调用 ListWebSpaces 和 ListWebSites 并输出结果。

```
static void Main(string[] args)
{
    X509Certificate2 certificate = GetStoreCertificate(ThumbPrint);
    List<string> myWebSpaces = ListWebSpaces(SubscriptionId, certificate,
    Version);
    foreach (var webSpace in myWebSpaces)
    {
        Console.WriteLine(webSpace);
        List<webSiteBasics> siteBasics = ListWebSites(SubscriptionId,
        certificate, webSpace, Version);
        foreach (var site in siteBasics)
        {
            Console.Write("\t");
            Console.WriteLine(site.Name);
            Console.WriteLine("\t\t" + "SiteMode:" + "\t" + site.SiteMode);
        }
    }
}
```

（9）下面是一个实例输出结果：

```
eastasiawebspace
        DebugChina
                SiteMode:       Basic
        clitestsite
                SiteMode:       Standard
        drumboy
                SiteMode:       Free
        hktestsitewzhao
                SiteMode:       Shared
eastuswebspace
        eutest
                SiteMode:       Standard
```

3.7　使用管理库管理网站

使用 Microsoft Azure 服务管理 API，必须自己构造 HTTP 请求的头部信息和内容。使用 Microsoft Azure 管理库，可以从构建 HTTP 请求和解析 HTTP 响应的具体工作中解放出来，而只需关注应用本身。Microsoft Azure 管理库是一个跨平台的.NET 库，提供了对 REST API 的封装。在底层，Microsoft Azure 管理库调用 Microsoft Azure 服务管理 REST API。它具有如下特点：

（1）支持可移植类库（PCL）。可以在.NET Framework 4.5 应用、Windows Phone 8、Windows 商店应用以及 Silverlight 应用程序中轻松调用 Microsoft Azure 服务管理库。

（2）它以一组 NuGet 软件包发布，降低彼此之间的依赖关系，以简化版本。

（3）支持异步操作。

（4）提供错误处理、跟踪、配置和 HTTP 管道处理等通用功能。

（5）易于测试。

（6）基于 HttpClient 和 Json.NET。

Microsoft Azure 服务管理库提供了丰富的 Microsoft Azure 服务管理接口，从而使用户能够自动化管理、配置和部署 Microsoft Azure 服务。

3.7.1　应用实例

在下面的例子中，通过 WAML 编写程序来重启所有网站。使用 WAML 同样要求使用证书。在下面的例子中，继续使用之前创建的证书。

（1）打开 Visual Studio，新建一个 Visual Studio 命令行工程，命名为 WALMDemo。

（2）安装 WAML 扩展包。

单击“工具”，选择 Library Package Manager→Package Manager Console。在控制台中运行以下命令：

```
PM> Install-Package Microsoft.WindowsAzure.Management.WebSites -Pre
```

该命令会安装 WAML 通用的扩展库以及 WebSites 扩展库。

（3）打开 Program.cs 文件，定义查找证书的函数，该函数在当前用户的证书库中根据指定的证书指纹来查找证书。之前运行 makecert 命令生成两个证书，公有密钥已经上传到 Azure 管理门户网站。现在使用的是包含私有密钥的证书。

```
private static X509Certificate2 GetStoreCertificate(string thumbprint)
{
    foreach (var location in locations)
    {
        X509Store store = new X509Store("My", StoreLocation.CurrentUser);
        try
        {
            store.Open(OpenFlags.ReadOnly | OpenFlags.OpenExistingOnly);
            X509Certificate2Collection certificates = store.Certificates.
            Find(
                X509FindType.FindByThumbprint, thumbprint, false);

            if (certificates.Count >= 1)
            {
                return certificates[0];
            }
        }
```

```
            finally
            {
                store.Close();
            }
        }
```

（4）在 Program.cs 文件中，在 main 函数中加入如下代码。与调用 REST API 相同，调用 WAML 需要提供证书和订阅 ID，请用证书指纹和订阅 ID 取代下面的 xxxxxx。

```
    static void Main(string[] args)
    {
        var myCert = GetStoreCertificate("xxxxxx");
        var credential = new CertificateCloudCredentials("xxxxxx);
        WebSiteListParameters siteProperties = new WebSiteListParameters();
        siteProperties.PropertiesToInclude.Add("Name");
        WebSiteManagementClient client =CloudContext.Clients.CreateWebSiteManagement
        Client(credential);
        WebSpacesListResponse webSpaces = client.WebSpaces.List();
        foreach (var webSpace in webSpaces.WebSpaces)
        {
           WebSpacesListWebSitesResponsesites=client.WebSpaces.ListWebSites (webSpace.
           Name, siteProperties);
            foreach (var site in sites.WebSites)
            {
                client.WebSites.Restart(webSpace.Name, site.Name);
            }
        }
    }
```

相比直接调用 Azure 管理 REST API，WAML 只需很少的代码即可实现相应的功能，代码更简洁。基于 WAML，开发人员不需要关注如何构造 HTTP 请求以及如何解析 HTTP 返回的内容。

3.8　参考文献与扩展阅读

1. Service Management REST API Reference

Azure 官方文档，介绍了如何通过服务管理 REST API 来管理部署在 Azure 中的资源。

http://msdn.microsoft.com/en-us/library/azure/ee460799.aspx

2. Web Sites Management REST API Reference

Azure 官方文档，介绍了如何通过服务管理 REST API 来管理部署在 Azure 中的资源，包括简介、API 版本、认证、网站资源结构以及对应的操作。

http://msdn.microsoft.com/en-us/library/azure/dn166981.aspx

3. Getting Started with the Windows Azure Management Libraries for .NET

Brady Gaster 博客文章，介绍了 Windows Azure Management Libraries for .NET，包含简单的实例代码。

http://www.bradygaster.com/post/getting-started-with-the-windows-azure-management-libraries

4. Microsoft Azure Web Sites Management Library 4.0.0

Microsoft Azure Web Sites Management Library 4.0.0 官方网站。可以通过 WAML 管理、部署、配置和扩展网站。

http://www.nuget.org/packages/Microsoft.WindowsAzure.Management.WebSites/

5. Azure command-line tool for Mac and Linux

Azure 官方文档，通过 Azure 命令行工具，可以创建、部署、管理 Azure 中的资源。该工具基于 Node.js，可以在 Mac 和 Linux 操作系统上运行。

http://azure.microsoft.com/en-us/documentation/articles/command-line-tools/

6. How to install and configure Azure PowerShell

Azure 官方文档，Azure PowerShell 提供了用于管理 Azure 资源的 PowerShell 命令行。可以手工执行这些命令行或者通过编程的方式创建一个自动化管理资源的脚本。

http://azure.microsoft.com/en-us/documentation/articles/install-configure-powershell/

第 4 章　Azure 网站应用开发框架

Microsoft Azure 网站是一个开放、灵活的平台。Microsoft Azure 网站原生支持各种主流应用框架和语言，包括 ASP、ASP.NET、PHP、Node.js、Phython 和 Java 等。在 Azure 网站中，有超过半数的客户运行 PHP 和 Node.js 等开源应用。在本章中，将介绍各种应用框架在 Azure 中的实现。

Microsoft Azure 网站无缝支持 Visual Studio。使用 Visual Studio 开发基于 Azure 网站的 Web 应用与开发本地 Web 应用完全相同。可以在 Visual Studio 中创建、部署、管理和配置 Azure 网站，并使用 Visual Studio 远程调试和诊断。Microsoft Azure 网站同时集成了 Visual Studio Online 在线编辑网站，可以在线编辑代码。

如果没有安装 Visual Studio，可以使用免费的 Visual Studio Express 工具快速开发 ASP.NET 网站、Web API 并部署到 Microsoft Azure 网站。Visual Studio Express 工具可以生成基于标准的高质量网站，内置了对当今 Web 标准的支持、完善的 CSS 设计功能和可视化诊断工具。

PHP 或者 Node.js 开发人员可以使用 WebMatrix。Web Matrix 是一个完全免费的、轻量级的、面向云的 Web 开发工具。使用 WebMatrix，可以轻松创建、发布和维护部署在 Azure 上的网站。WebMatrix 支持 ASP.NET、PHP、Node.js 和 HTML5 网站，并支持最新的 Web 标准（CSS3、HTML5）和流行的 JavaScript 库，比如 JQuery。除此之外，WebMatrix 同时支持源代码版本控制，比如 Git 和 TFS。

可以通过前面介绍的 Web 平台安装程序下载并安装 Visual Studio Express 和 WebMatrix。

同时，Microsoft 推出了基于 Visual Studio 的 PHP 和 Node.js 扩展，开发人员可以使用 Visual Studio 开发 PHP 和 Node.js。

本章首先介绍 Azure 网站的文件目录结构，以及如何使用 FTP 访问 Azure 网站文件。之后介绍如何在 Visual Studio 中集成 Azure 网站。本章的主要篇幅用于介绍 3 种主流应用开发语言与框架在 Azure 网站上的实现。

4.1　Azure 网站文件目录结构

4.1.1　Azure 网站文件目录介绍

Azure 网站文件根目录（d:\home）下有 3 个目录：Data、LogFiles 和 site 目录。其中，

Data 目录下主要保存 Web 作业的日志。Logfiles 用于保存应用日志、网站日志以及各种错误日志。网站的文件保存在 site\wwwroot 目录下，它是网站的根目录。

下面是主要目录及其作用描述。

```
Data
        Jobs
            Triggered
                MyTriggeredJob1
                    20131112101559
                        output.log          //WebJob 输出
                        error.log           //WebJob 错误信息
                        status              //WebJob 执行状态
            Continuous
                MyContinuousJob1
                    job.log                 //WebJob 日志
                    status                  //WebJob 执行状态

LogFiles
        Application
            <pid>-<ticks>-<instance>.txt  //应用程序诊断日志
        DetailedError
            ErrorPage####.htm             //详细的错误消息
        Git
            trace
                trace.xml                 //Git 部署时产生的跟踪信息
                <instance>-<guid>.txt     //kudu 相关的信息
            deployment
                <instance>-<guid>.txt     //与部署相关的信息
        http
            RawLogs
                <logfile>.log             //IIS 日志（每 60s 更新一次）
        W3SVC#####
            fr####.xml                    //IIS 失败请求跟踪
            freb.xsl
site
        wwwroot
            hello.htm                     //网站内容
        repository                        //网站源代码管理存储库
            .git
                HEAD、索引和其他.git 文件
        deployments
            [commit id 1]
                log.xml                   //部署日志，与管理门户显示的信息类似
                status.xml                //部署状态（成功/失败）
                manifest                  //部署的文件列表
            [commit id 2]
                ...
        .ssh
            Config                        //配置
            id_rsa                        //私有密钥
```

```
        known_hosts                    //已知的主机列表
```

4.1.2　通过 FTP 访问 Azure 网站文件系统

Azure 网站提供了 FTP 功能，可以通过各种 FTP 客户端轻松访问 Azure 网站文件。下面的步骤演示了通过 FileZilla FTP 客户端访问网站文件。

4.1.2.1　下载发布配置文件

发布配置文件包含访问 FTP 的用户凭据，可以通过管理门户网站下载。

（1）登录到 Microsoft Azure 管理门户网站。

（2）选择要部署的站点，如图 4-1 所示，在“快速开始”页面单击“发布应用程序”下的“下载发布配置文件”

（3）或者，在页面顶部导航栏，单击“仪表板”。

（4）如图 4-1 所示，在“速览”下面，可以选择“下载发布配置文件”。

图 4-1　下载网站发布配置文件

发布配置文件是一个 XML 文件，包含 FTP 相关的配置和凭据。在发布配置文件中，如下所示，可以找到登录 FTP 必要的用户凭据，包括 FTP 主机名称、用户名称和密码。用户名称默认为 sitename\$sitename。

注意：该用户拥有 FTP 站点的完全控制权，可以修改、添加和删除文件及目录。

```
<publishProfile
 profileName="sitename - FTP"
 publishMethod="FTP"
 publishUrl=ftp://waws-prod-hk1-001.ftp.azurewebsites.windows.net/site/
wwwroot
 ftpPassiveMode="True"
 userName="sitename\$sitename"
 userPWD="wxfsaxJlhGe4NlkS0WiE37dacdmLLbE5RlrbqYHrJycymXbZ3Tcw8aiDuFF2"
 …
</publishProfile>
</publishData>
```

4.1.2.2　使用 FileZilla 访问 Azure 网站文件系统

（1）下载并安装 FileZilla。FileZilla 是一个免费开源的 FTP 客户端，可以通过 https://filezilla-project.org/网站下载。

（2）运行 FileZilla，然后选择 File→Site Manager。新建一个站点，如图 4-2 所示，给定主机名称、用户和密码。然后单击 Conneet。

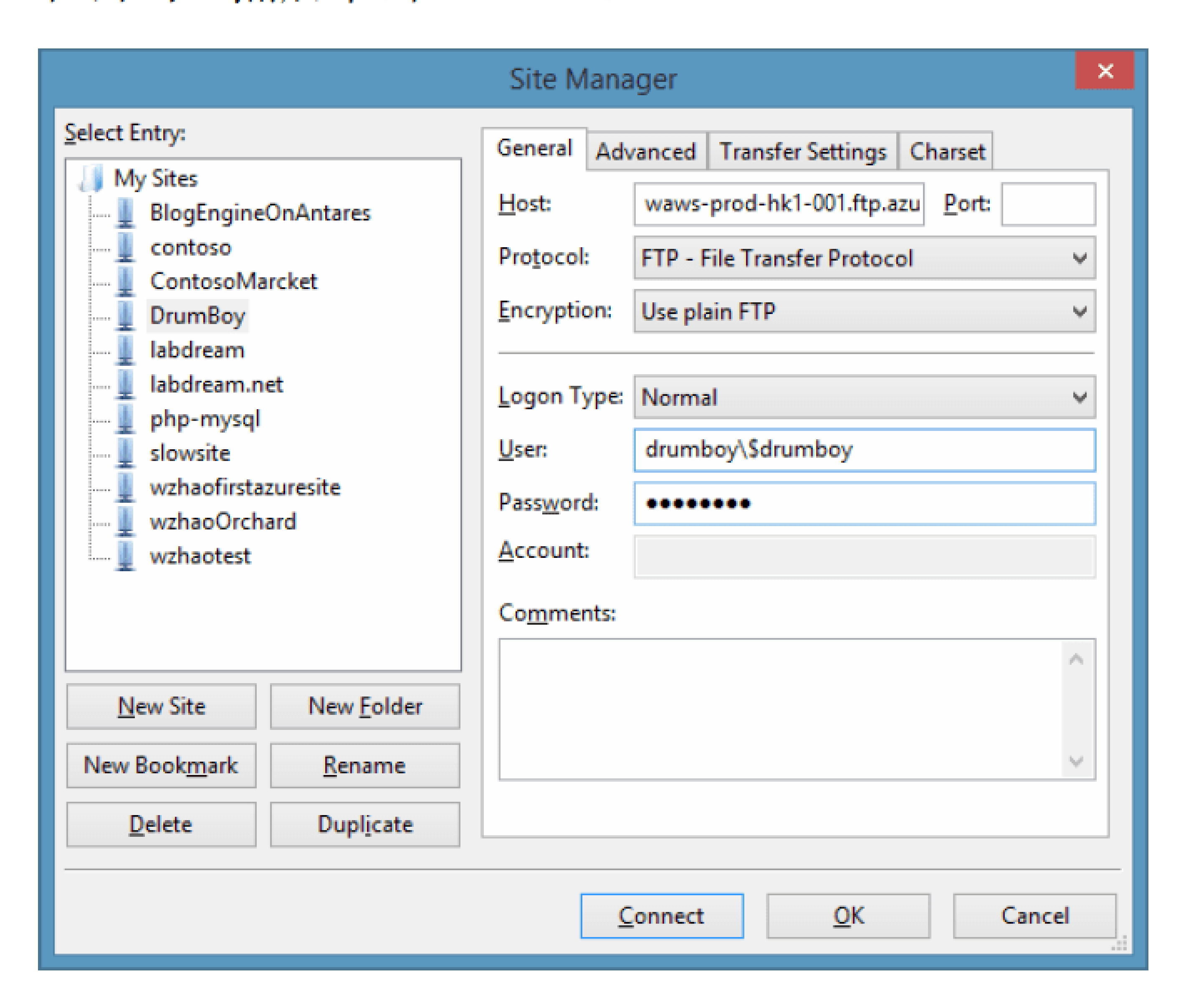


图 4-2　FileZilla 站点管理器

（3）连接成功后，如图 4-3 所示，即可在 FileZilla 中查看和修改网站文件。

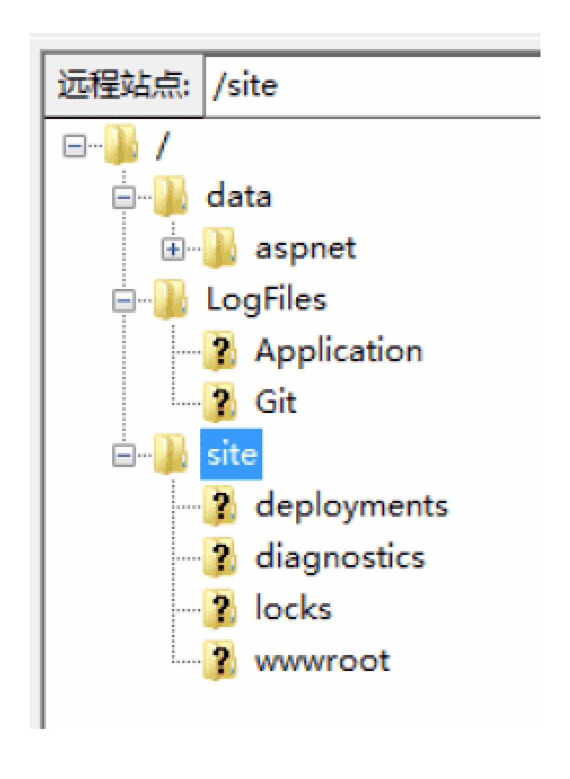


图 4-3　Azure 网站文件目录

4.2　在 Visual Studio 中集成 Azure 订阅

Visual Studio 中可以无缝集成 Azure 订阅。开发人员可以在 Visual Studio 中方便地管理部署在 Azure 中的资源。本节介绍如何将 Azure 订阅集成到 Visual Studio 中。

4.2.1.1　连接到全球 Azure 环境

（1）安装 Microsoft Azure SDK 和 Microsoft Azure Tools for Visual Studio。

该工具为 Microsoft Visual Studio 的 Microsoft Azure 扩展。通过该工具，可以在 Visual Studio 2012/2013 中创建、配置、开发、调试、运行、打包和部署高扩展性的 Microsoft Azure Web 应用程序和服务。如图 4-4 所示，可以通过 Web 平台安装程序来安装这两个软件。

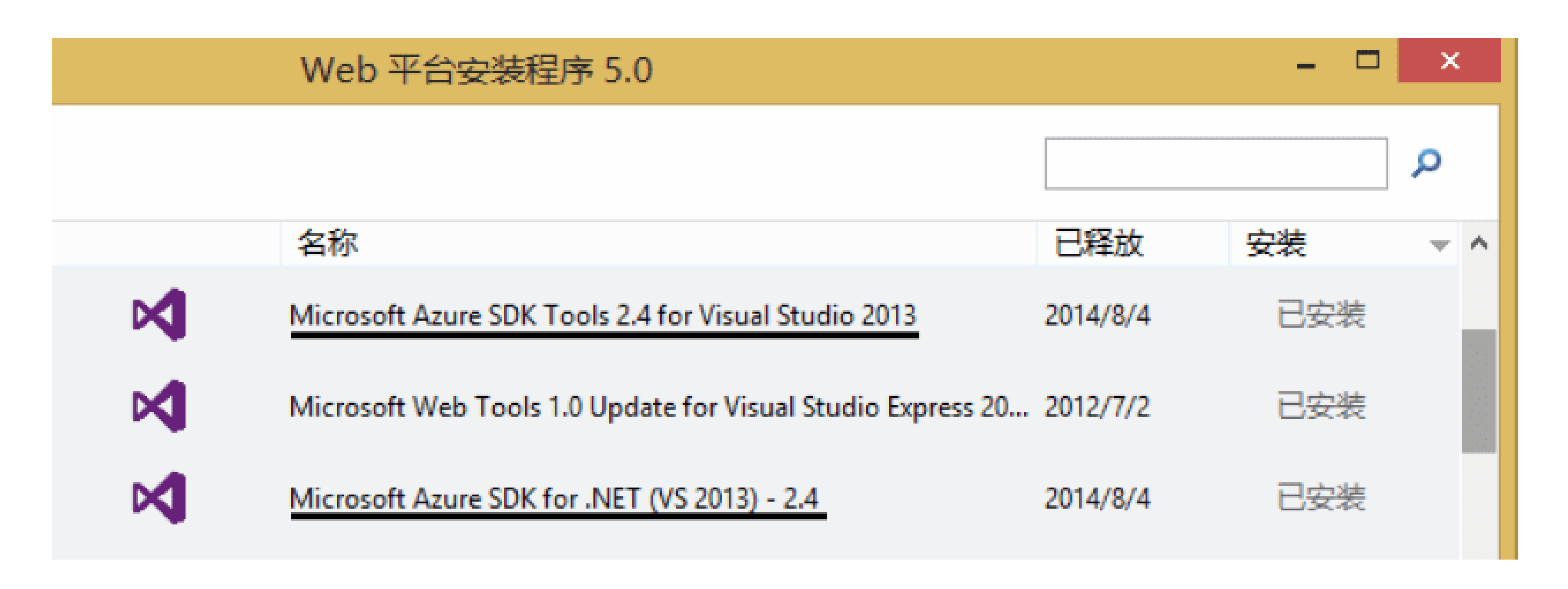


图 4-4　安装 Azure SDK

（2）运行 Visual Studio 2013 或者 Visual Studio Express for Web 2013。下面以 Visual Studio 2013 为例，Visual Studio Express 2013 for Web 与此类似。

（3）在服务器资源管理器中，如图 4-5 所示，右击 Azure，选择“连接到 Microsoft Azure”。

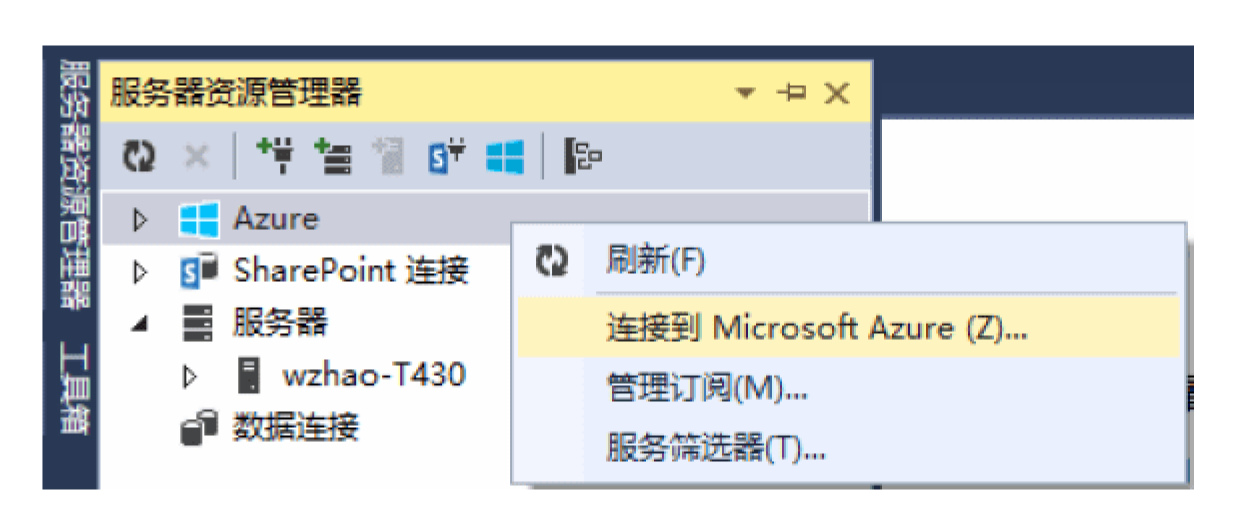


图 4-5　连接到 Azure 订阅

（4）此时，弹出“登录到 Microsoft Azure”窗口，输入 Azure 订阅邮件地址和密码，连接到 Azure，如图 4-6 所示。



图 4-6　登录到 Azure

（5）登录后，服务器资源管理器会连接到 Azure 同步订阅信息。同步完成后，会看到

已有的网站、数据库和移动服务。右击 WebSites，选择 Create New Site，如图 4-7 所示。

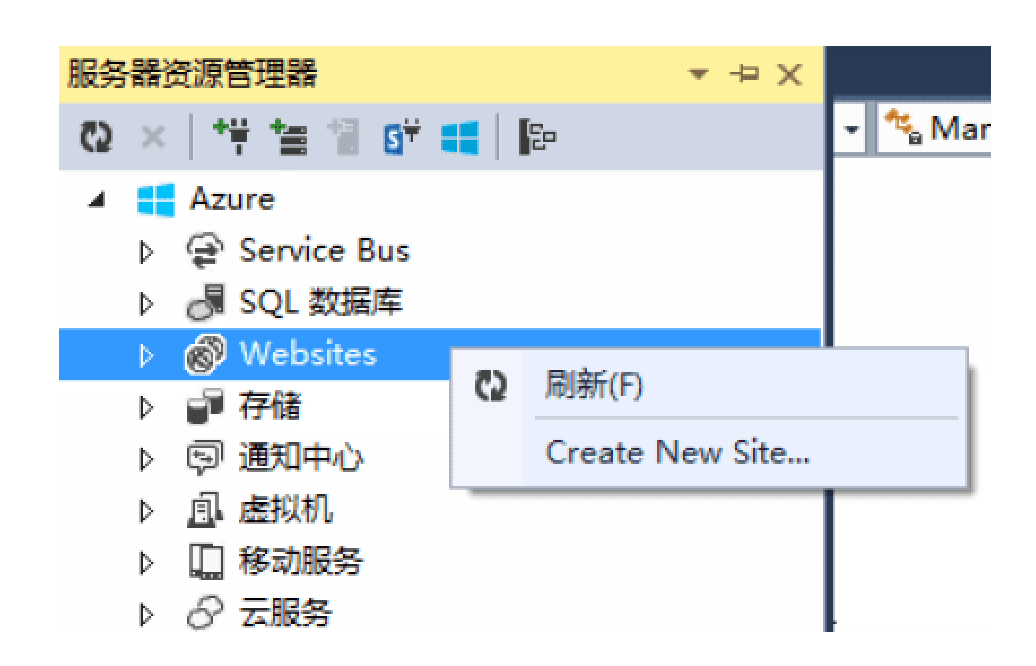


图 4-7　创建新网站

（6）如图 4-8 所示，指定网站名称，数据中心，单击“创建”按钮。

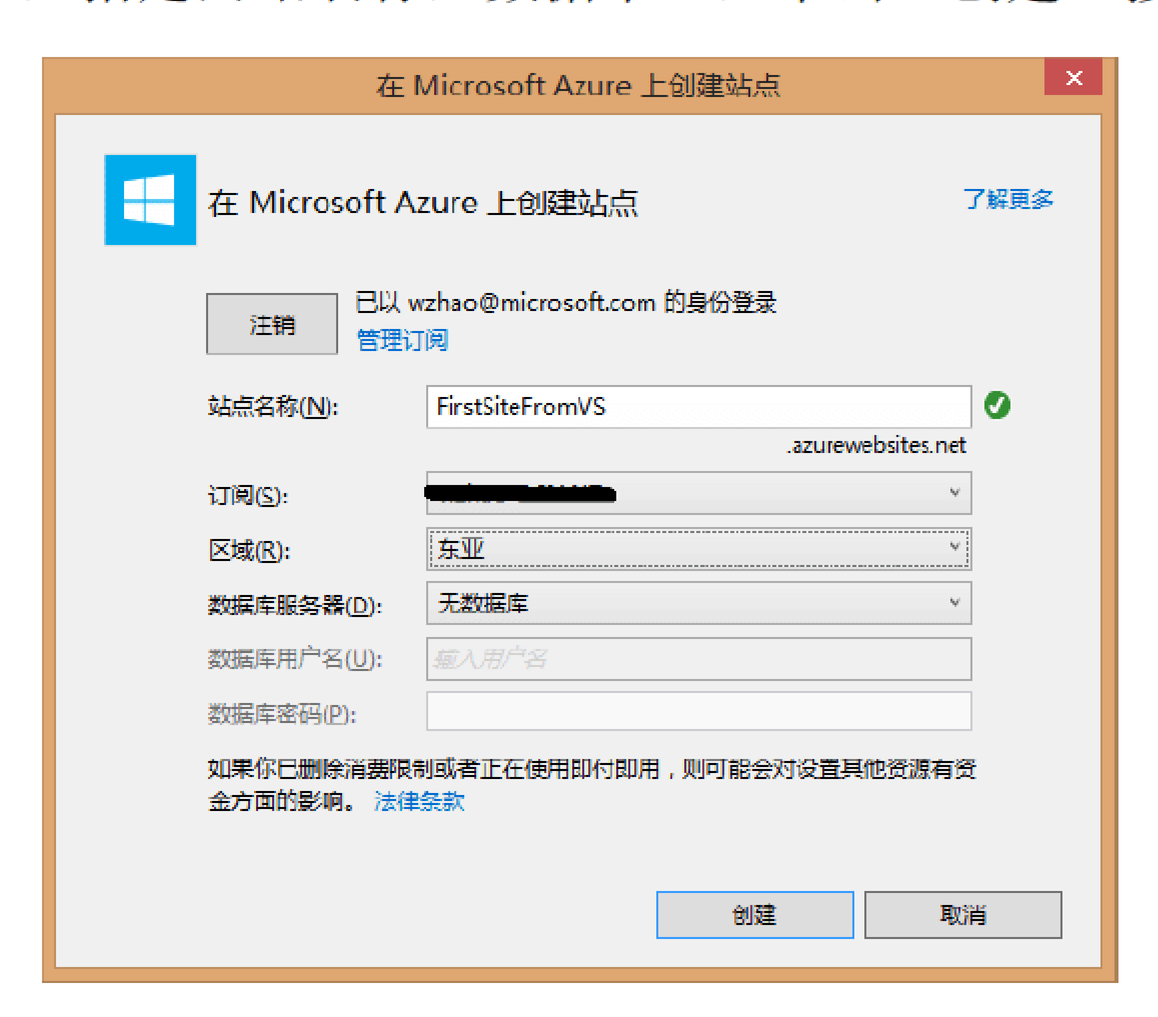


图 4-8　指定网站名称

（7）稍等片刻后，网站即创建成功。此时回到服务器资源管理器中，可以看到新创建的网站。右击该网站，选择“在浏览器中查看”。如图 4-9 所示，此时，网站会在浏览器中打开。

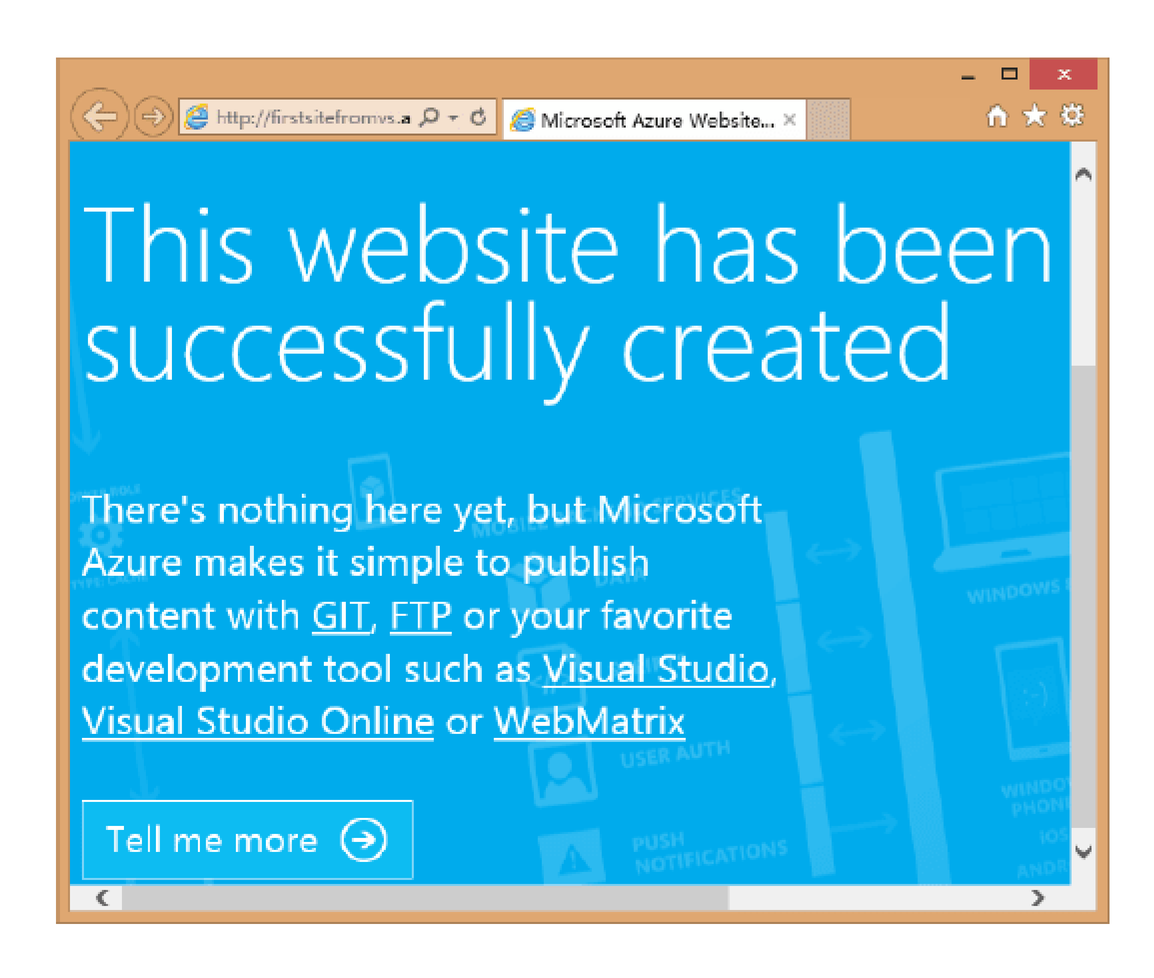


图 4-9　新建的空网站

4.2.1.2　连接到中国 Azure 环境

因为 Visual Studio 中的服务器资源管理器自动连接到全球 Azure 订阅，要连接到中国 Azure 环境需要下面的步骤：

（1）访问下面的链接下载中国区 Azure 发布订阅文件：

http://go.microsoft.com/fwlink/?LinkID=301776

（2）在 Visual Studio 的服务器资源管理器中，右击 Azure，选择“管理订阅（M）…”命令，如图 4-10 所示。

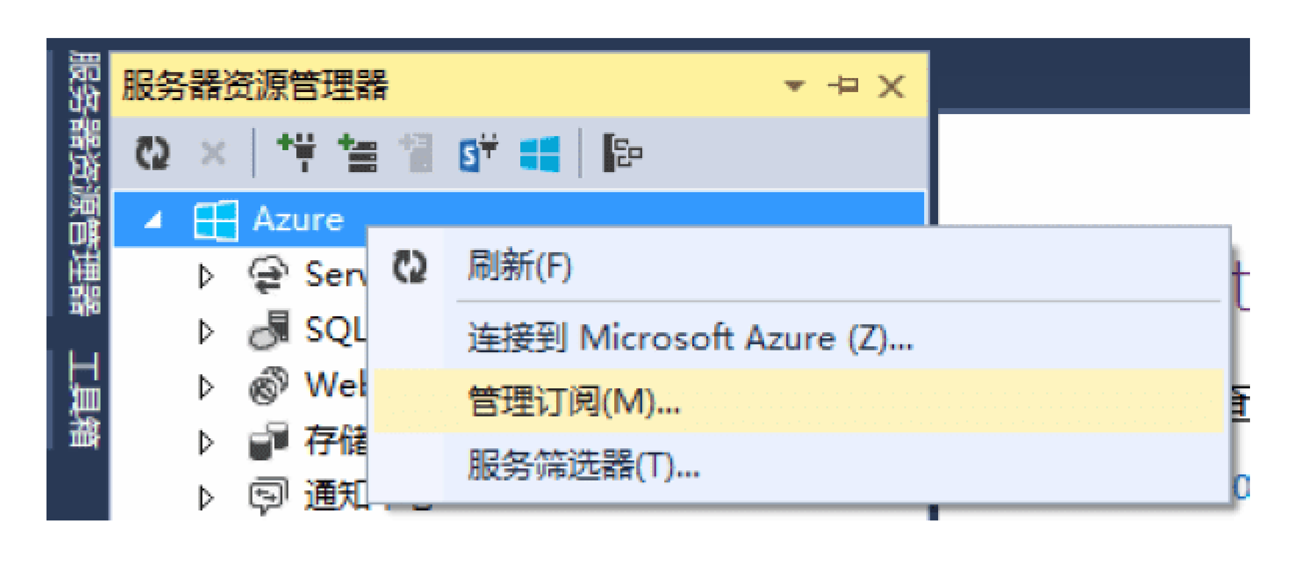


图 4-10　管理订阅

（3）在“管理 Microsoft Azure 订阅”窗口中，选择“证书”，然后单击“导入”按钮，如图 4-11 所示。在“导入 Microsoft Azure 订阅”对话框中选择第一步下载的订阅文件，最后单击“导入”按钮。

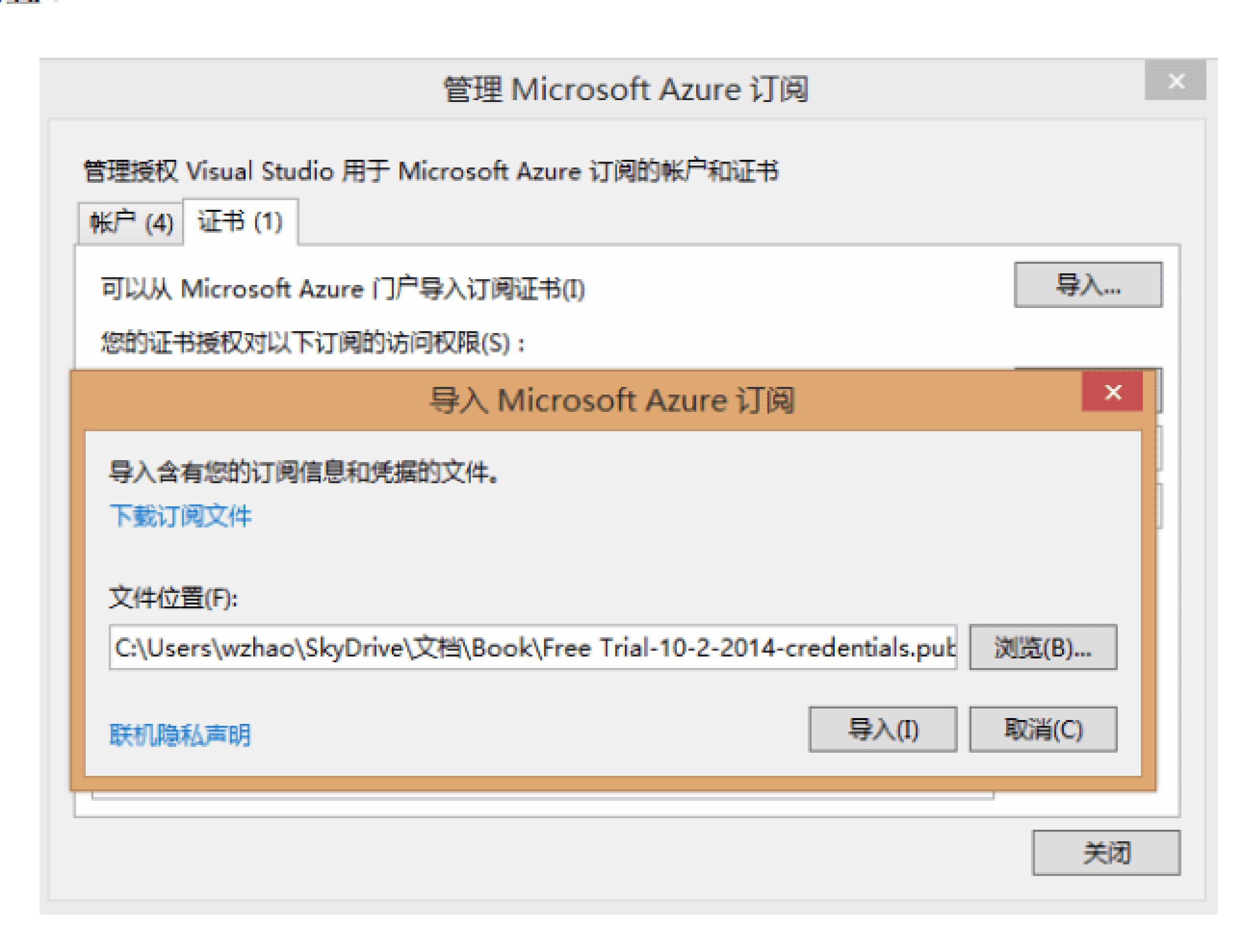


图 4-11　导入 Microsoft Azure 订阅

4.3　Azure 网站上的 ASP.NET

Azure 网站原生支持 ASP.NET，与任何本地开发环境并无任何不同。Azure 网站支持基于下列版本的.NET 应用：

- .NET Framework V3.5，包含.NET 2.0、3.0、3.5。

- .NET Framework V4.5，包含.NET 4.0.、4.5、4.5.1。

4.3.1 创建一个 Web 项目

（1）在 Visual Studio 2013 中选择 File→New→Project 命令。

（2）如图 4-12 所示，在 New Project 窗口中选择 ASP.NET Web Application，选择.NET Framework 4.5，命名为 MyFistAzureSite，单击 OK 按钮。

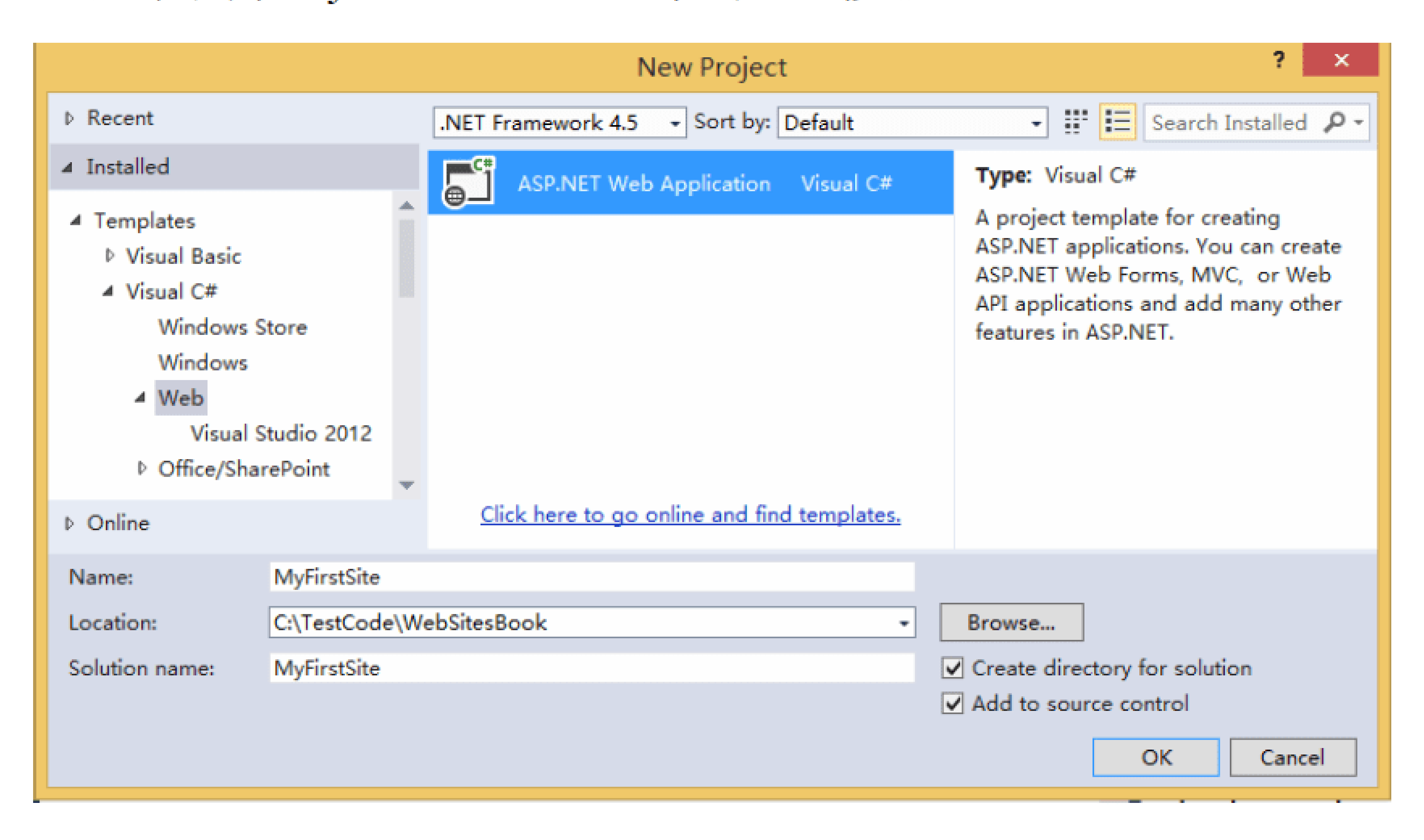


图 4-12 创建新的 ASP.NET 项目

（3）如图 4-13 所示，选择 Empty，单击 OK 按钮，创建空项目。

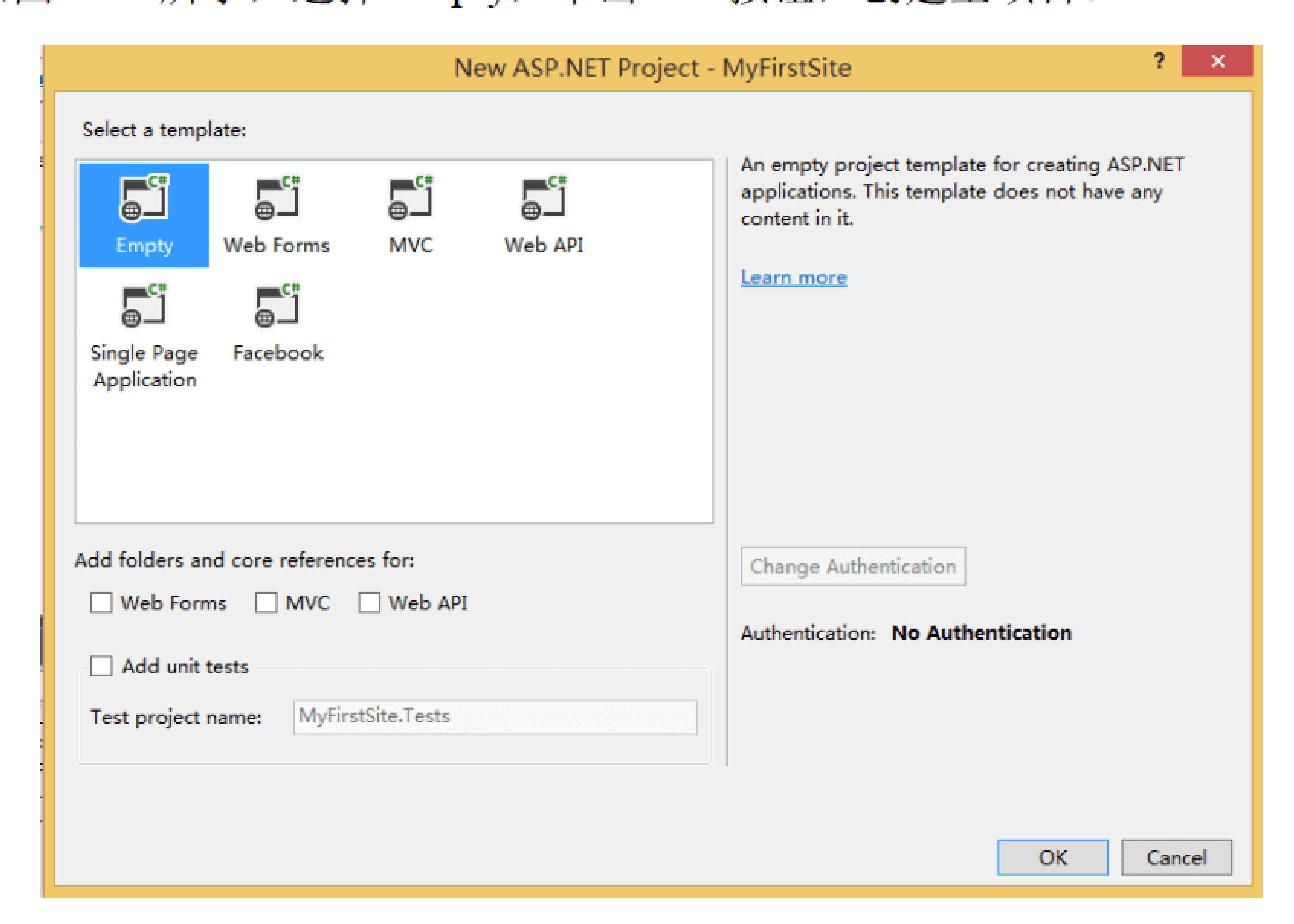


图 4-13 创建空的 ASP.NET 项目

（4）右击 MyFistAzureSite 项目，选择 Add→New Item 命令，如图 4-14 所示，选择 Web Form，命名为 default.aspx。

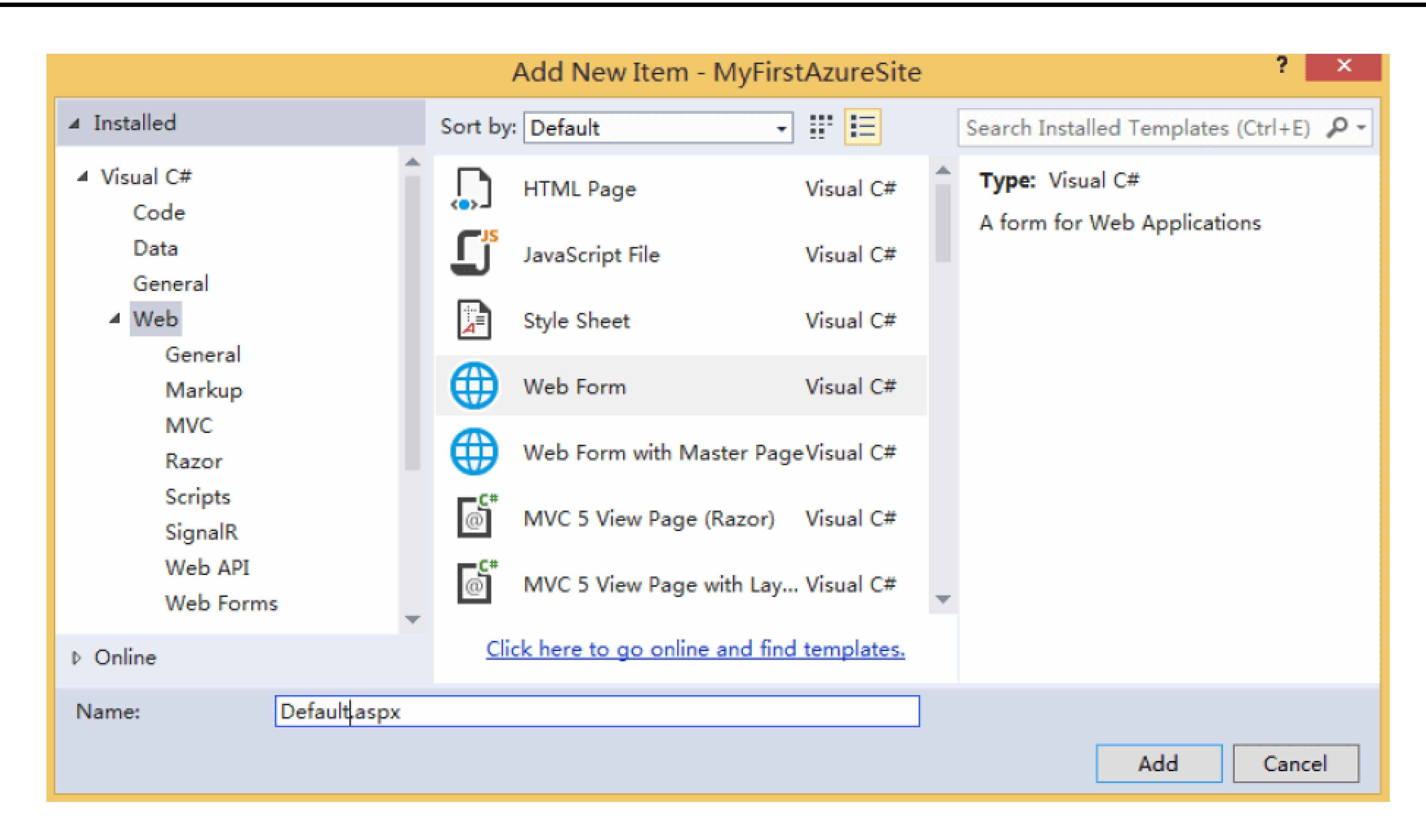


图 4-14　添加 Web Form

（5）在 Default.aspx.cs 文件中找到 Page_Load 函数，加入一行代码将网页的标题修改为服务器时间和时区名称，如下所示：

```
protected void Page_Load(object sender, EventArgs e)
{
    this.form1.InnerText = System.DateTime.Now.ToString() +
    "  (" + System.TimeZone.CurrentTimeZone.StandardName + ")";
}
```

4.3.2　将网站部署到 Azure 网站

（1）在解决方案资源管理器中，右击 MyFirstAzureSite 项目，选择“发布”命令。

（2）如图 4-15 所示，在“发布 Web”对话框中，单击“导入”按钮，导入发布配置文件。

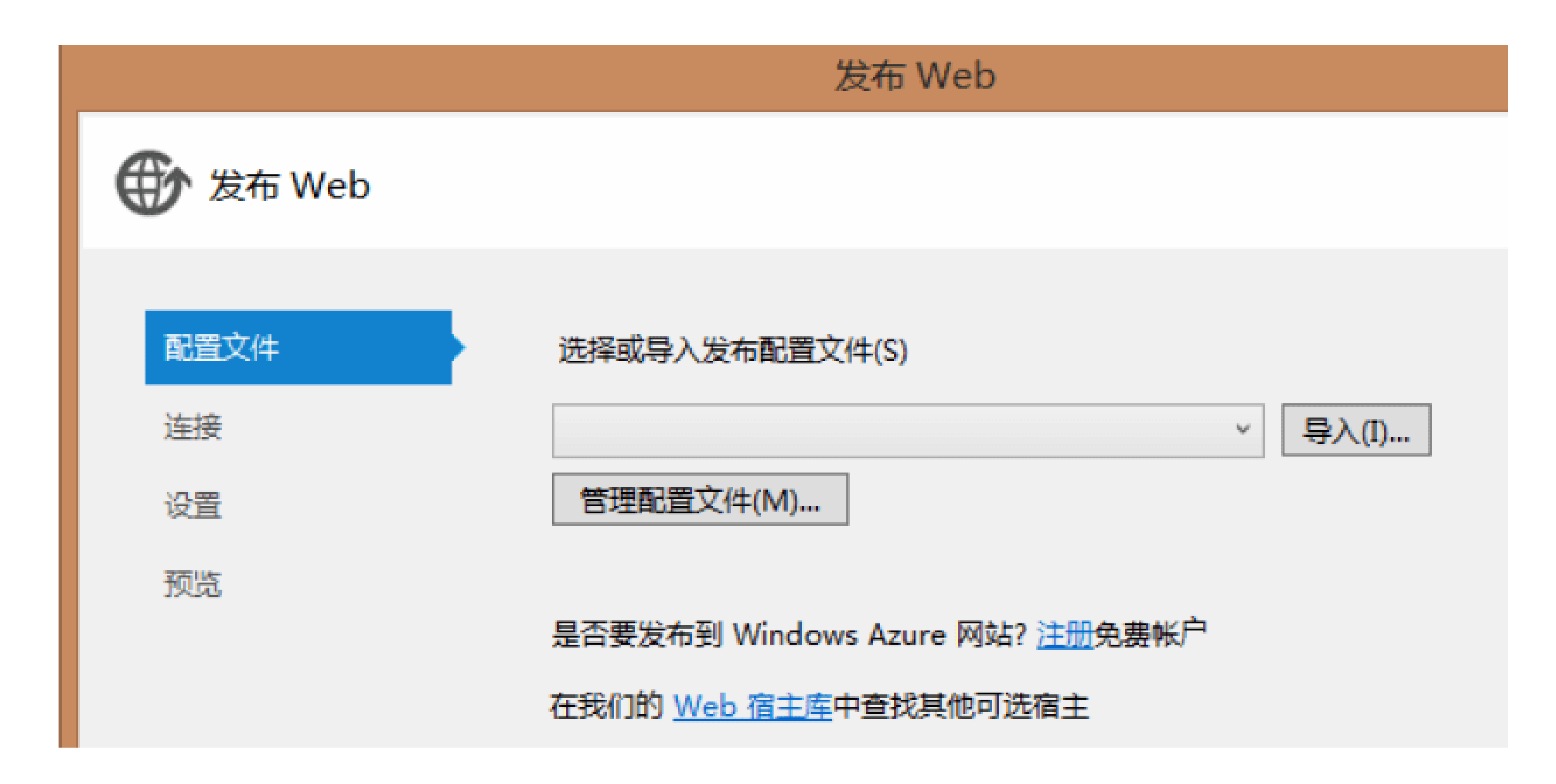


图 4-15　“发布 Web”对话框

（3）如图 4-16 所示，在“导入发布配置文件”对话框中，选择“从 Microsoft Azure 网站导入”。单击下拉列表，选择前面创建的网站。

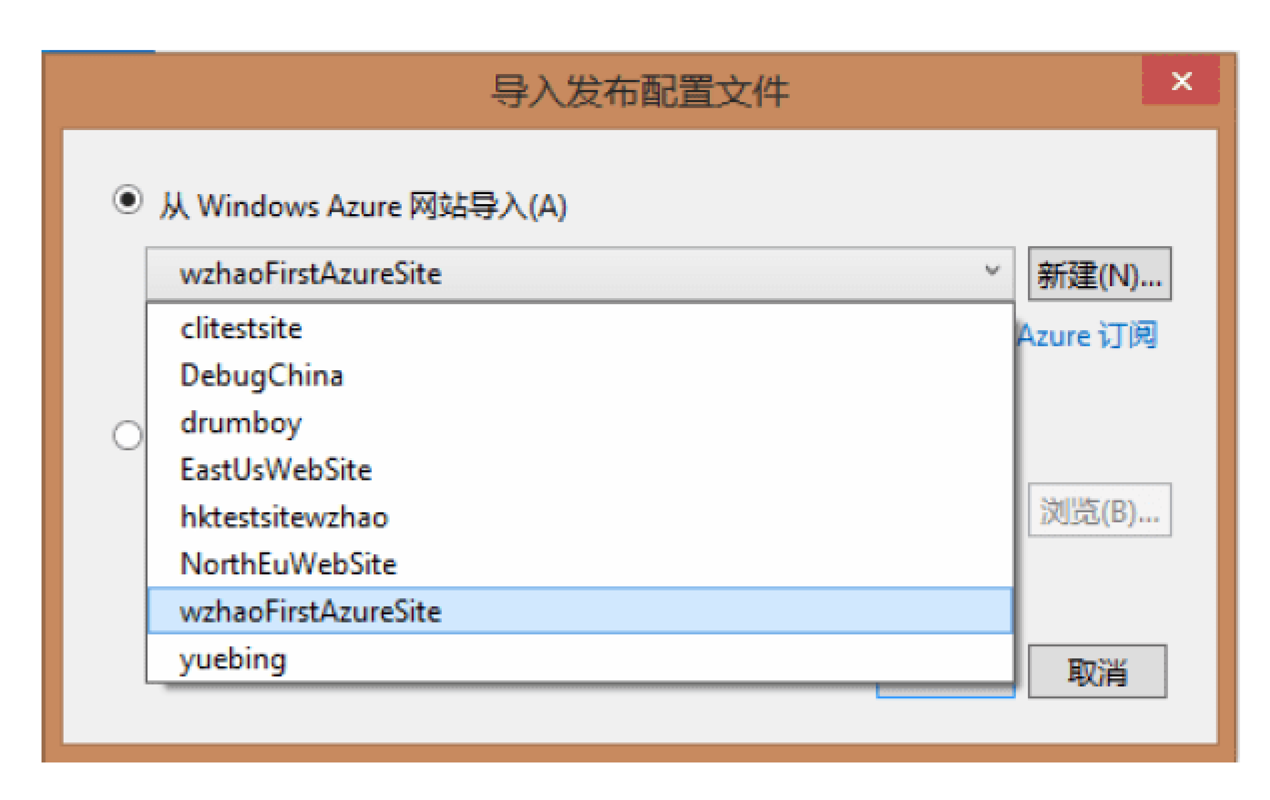


图 4-16　导入发布配置文件

（4）单击 OK 按钮，此时 Visual Studio 会自动下载选中的网站的发布配置文件。下载完成后，如图 4-17 所示，会看到网站发布配置的选项。单击“验证连接”按钮，Visual Studio 自动验证网络连接。验证通过后，会看到绿色的对号。

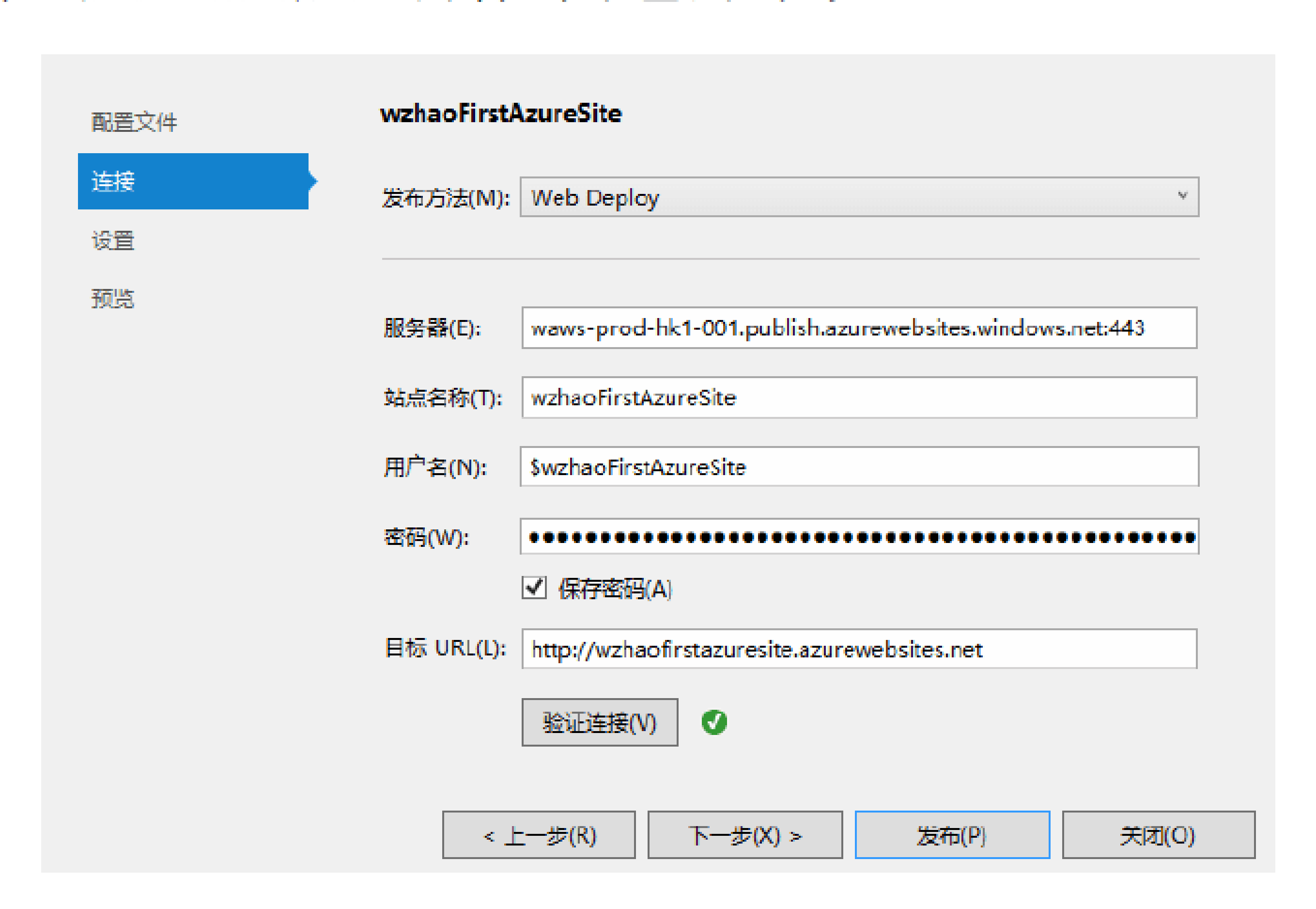


图 4-17　验证连接

（5）单击“下一步”按钮可以进一步设置相关的文件发布选项和数据库选项。

（6）单击“发布”按钮，在 Visual Studio 的输出窗口，会看到网站发布的具体信息。如图 4-18 所示，在 Web 发布活动窗口也可以看到类似信息。

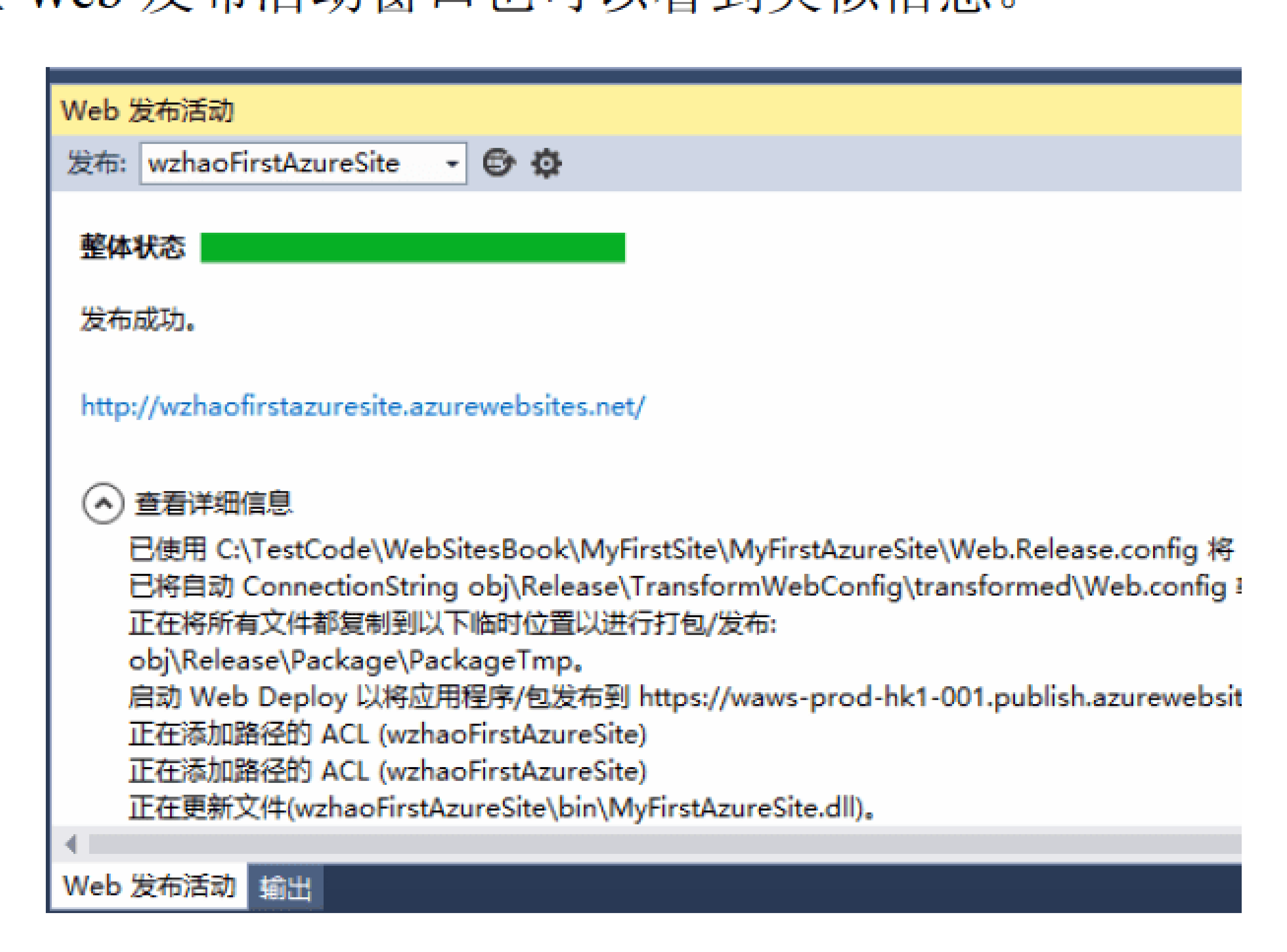


图 4-18　发布结果

（7）网站发布成功后，Visual Studio 会自动打开默认浏览器浏览网页。如图 4-19 所示，网页显示服务器当前时间。如网页结果所示，不管 Azure 网站服务器在哪个数据中心，所有的 Azure 网站服务器都使用标准 UTC 时间。

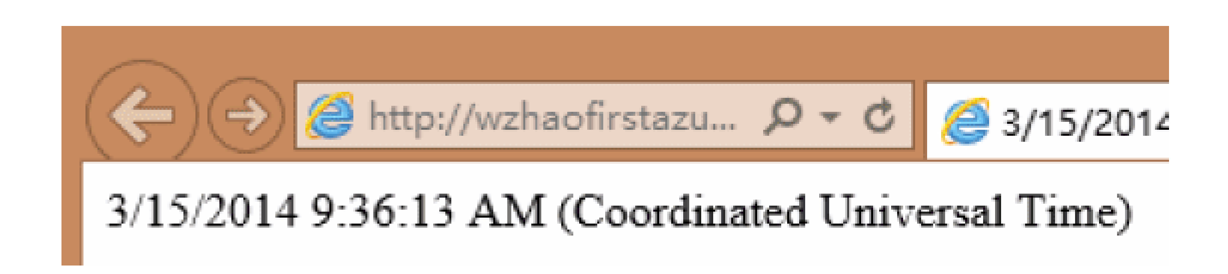


图 4-19　网站虚拟机采用 UTC 时间

4.3.3　Azure 网站中 ASP.NET 开发常见问题

Microsoft Azure 网站提供了几乎与本地开发运行完全相同的运行环境。同时，Visual Studio 也无缝支持 Azure 网站的开发。因此，绝大多数应用可以运行在 Azure 网站而无需任何改动。尽管如此，由于开发人员对于 Azure 网站环境的不熟悉，还是会遇到一些问题。下面是客户遇到最多也是论坛里面最常见的问题。

4.3.3.1　不能加载文件或程序集（Could not load file or assembly）

如果 ASP.NET 网站使用了第三方的程序集，可能会遇到该问题。在本地开发和运行环境中，第三方的程序集已经安装，所以 ASP.NET 应用可以加载这些程序集。但是在 Azure 网站中，并没有安装这些程序集。Microsoft Azure 网站提供了一个标准的、干净的运行环境。该环境只包含完整的.NET 2.0、.NET 3.0、.NET 3.5、.NET 4.0 和.NET 4.5 的安装，不包含任何的第三方程序集。因此，如果您的应用使用了第三方的程序集，在部署到 Azure 网站以后，可能会遇到下面无法加载程序集的错误：

Could not load file or assembly xxxxxx, Version=x.x.0.0, Culture=neutral, PublicKey Token =x xx xxxxxxxxxx or one of its dependencies. The system cannot find the file specified.

要解决该问题，只需如图 4-20 所示，在解决方案资源管理器中，将引用的第三方的程序集“复制本地”属性设置为 True。这样，Visual Studio 在部署的时候，会将该程序集部署到 Azure 网站的 bin 目录下。

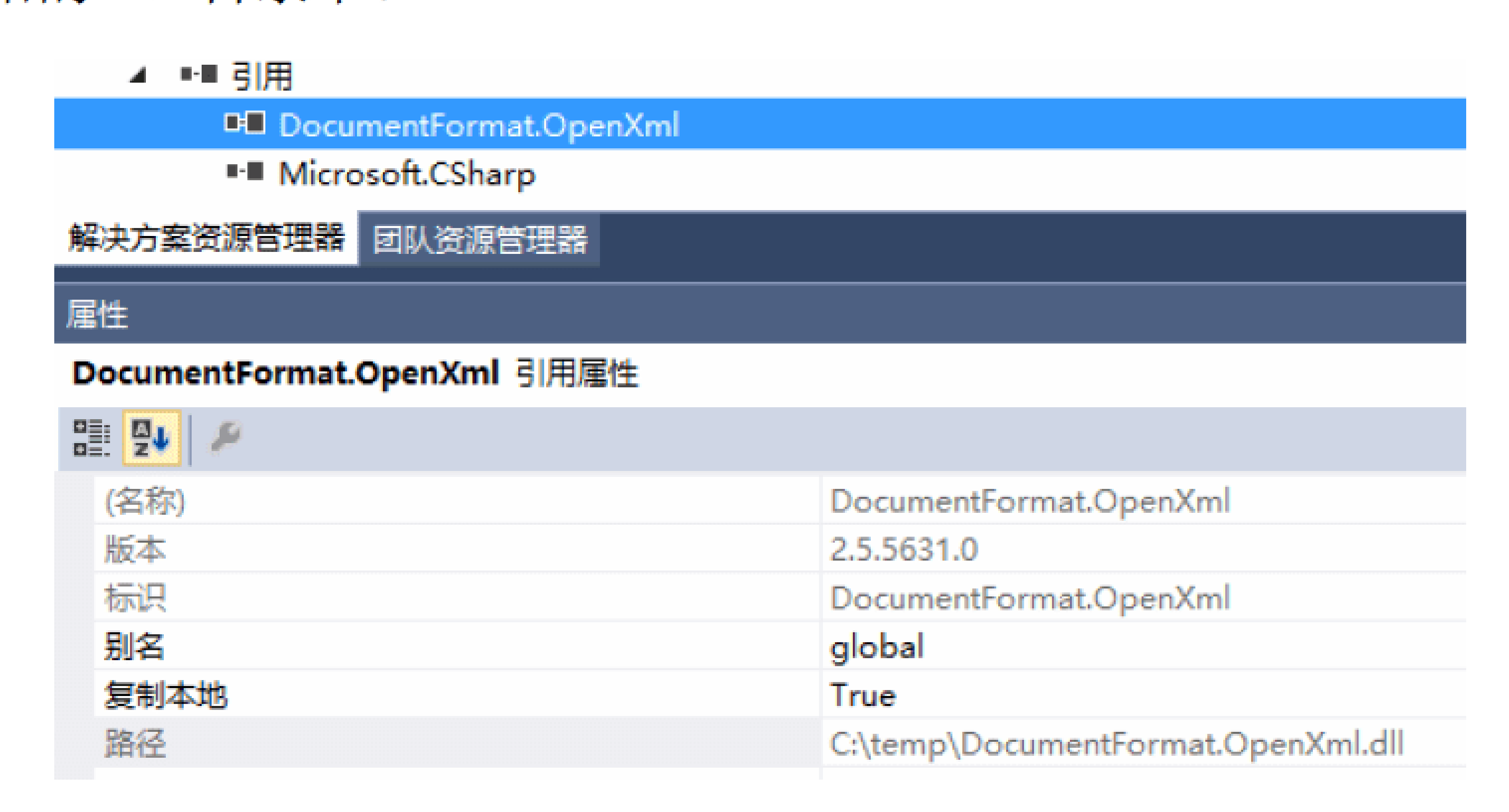


图 4-20　部署时将程序集复制到 Azure 网站

还有一个问题需要注意。Microsoft Azure 网站运行在 64 位操作系统，但是默认使用 32 位 IIS 工作进程。其原因是 32 位 IIS 工作进程可以满足绝大多数网站的需要，而且相比 64 位工作进程更节约资源，从而提高硬件效率。所以如果 ASP.NET 网站引用了只兼容 64 位的程序集，那么会看到下面的错误：

Exception message: Could not load file or assembly xxxxxx or one of its dependencies. An attempt was made to load a program with an incorrect format.

要解决该问题，必须将 Microsoft Azure 网站的平台设置为 64 位。注意，只有标准模式的网站支持 64 位平台。如图 4-21 所示，可以登录到 Microsoft Azure 管理门户网站，在配置页面指定使用 32 位还是 64 位平台。



图 4-21 选择平台

4.3.3.2 不能加载用户的配置文件（cannot load the user's profile）

在 Windows 系统中，在用户第一次登录到计算机时创建一个用户配置文件。用户配置文件包含以下信息：

（1）一个注册表配置单元。系统在用户登录时加载用户的注册表文件，并把它映射到 HKEY_CURRENT_USER 注册表键。用户的注册表配置单元保存用户的基于注册表的喜好和配置。

（2）一组存储在文件系统的用户配置文件夹。用户配置文件存储在配置文件目录中，每个用户一个单独的文件夹。应用程序和其他系统组件将用户的数据，如文件和配置文件等保存在用户配置文件夹。 Windows 资源管理器使用用户配置文件存放用户的桌面、开始菜单和文件夹等信息。

用户配置文件有以下优点：

（1）当用户登录到计算机时，系统会自动加载用户上次登录时使用的设置。

（2）当与其他用户共享一台计算机时，每个用户登录后系统显示用户的自定义桌面等。

（3）在用户配置文件中保存的设置不能被其他用户访问。一个用户的配置文件所做的更改不会影响其他用户或其他用户的个人资料。

（4）用户可以使用自定义的环境变量。

（5）用户可以使用单独的临时文件夹。如果加载用户配置文件，则临时文件夹为 c:\users\<username>\Appdata\local\temp，否则使用 c:\windows\temp 目录。

可以通过设置 loadUserProfile 来指定 IIS 的工作进程加载用户配置文件，该设置默认为 false。关于该配置的具体信息，请参考下面的文章：

http://www.iis.net/configreference/system.applicationhost/applicationpools/add/processmodel

Microsoft Azure 网站默认并不加载用户配置文件。如果应用需要加载用户配置文件，那么应用在部署到 Microsoft Azure 网站后可能会遇到问题。

比如，WIF 默认使用 DPAPI 来加密会话 cookie，DPAPI 需要加载用户配置文件，所以 WIF 应用部署到 Microsoft Azure 网站后会遇到问题。关于 WIF 的问题的具体信息和解决方案，请参考下面的文章：

http://www.cloudidentity.com/blog/2013/01/28/running-wif-based-apps-in-windows-azure-web-sites-4/

Azure 网站在最近的更新中解决了该问题，允许加载用户配置文件。如果应用需要加载用户配置文件，请遵循下面的步骤：

（1）登录到 Azure 管理门户网站。

（2）单击左侧导航栏中的网站图标。

（3）在右侧网站列表中选择需要加载用户配置文件的网站。

（4）在顶部导航栏，单击“配置”，打开网站的配置页面。

（5）定位到应用配置区域，如图 4-22 所示，设置应用变量 WEBSITE_LOAD_USER_PROFILE=1。

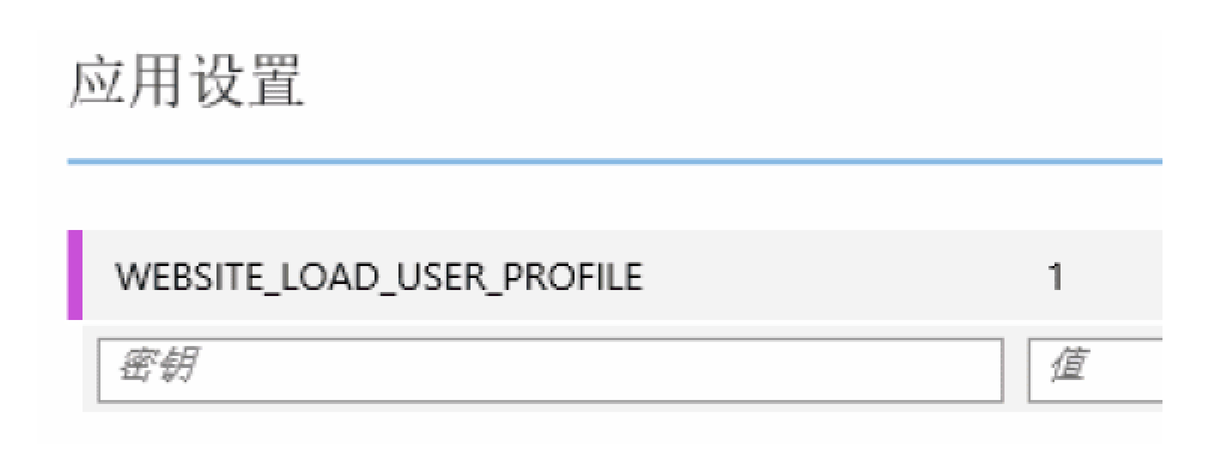


图 4-22　设置 WEBSITE_LOAD_USER_PROFILE

（6）单击底部命令栏的“保存”按钮。

（7）单击底部命令栏的“重新启动”按钮，重启网站。

4.3.3.3　负载均衡问题

如果网站运行了多个实例，那么 Microsoft Azure 的前端服务器自动为网站提供负载均衡功能。Microsoft Azure 服务器使用的负载均衡算法是当前最少请求算法，它把客户请求分发到当前正在执行的请求数最少的机器。如果网站需要运行多个实例，在部署到 Microsoft Azure 网站之前，需要考虑以下问题。

1. 会话（Session）

ASP.NET 支持 4 种保存会话的方式：

（1）保存在 IIS 工作进程内存中（In-Proc）。

（2）保存在会话状态服务中（Session State Service）。

（3）保存在 SQL 服务器中。

（4）自定义会话服务模块（Session Provider）。

In-Proc 选项不能用于多实例的负载均衡环境中。另外，Microsoft Azure 网站不支持会话状态服务。用户当然希望使用 SQL Azure 来保存会话状态以获得更好的性能，但是，默

认的 SQL 会话服务模块不支持 SQL Azure。对于 Azure 网站，可以使用 Azure 缓存服务来保存会话，具体请参考下面的文章：

http://msdn.microsoft.com/en-us/library/windowsazure/gg185668.aspx

另外，可以安装单独的 Microsoft.AspNet.Providers，该软件包支持 SQL Azure：

https://www.nuget.org/packages/Microsoft.AspNet.Providers/2.0.0

2. 应用程序状态

ASP.NET 应用程序状态（application state）是一个数据存储方法。应用程序状态存储在服务器 IIS 工作进程的内存中，比在数据库中存储和检索信息的速度更快。会话状态应用于单个用户会话，应用程序状态适用于所有用户和会话。因此，应用程序状态经常用于存储少量被所有用户共享的并经常使用的数据。

由于应用程序状态存储在服务器 IIS 工作进程的内存中，所以应用程序状态不能在多个实例之间共享。因此如果应用使用了应用程序状态，在运行多个 Microsoft Azure 网站实例时，各个实例之间的内容不同步，应用可能会遇到问题。可以将应用程序状态保存在 Microsoft Azure 缓存服务中。

4.3.3.4 安全沙漏

Microsoft Azure 网站是一个多租户的环境，多个用户的网站会同时运行在一台工作机器上。基于安全考虑，每个用户的网站是运行在一个称为沙漏的安全上下文环境中。沙漏直接互相隔离，保证用户直接数据隔离。另外，在 Microsoft Azure 网站上有些 API 是被禁止的。比如，GDI API 是一个被禁止的例子，另外，应用也不允许监听服务器的任何网络端口。

4.3.3.5 使用安全证书

在第 3 章，介绍了如何通过管理门户网站上传安全证书，并使用安全证书进行传输加密（HTTPS）。除了传输加密之外，很多关键应用要求使用安全证书进行数据加密、消息加密和数字签名等，比如：

- 保存用户密码。很多 Web 站点采用表单（form）认证，通常这些应用将用户名称与密码保存到数据库。一个安全的站点通常将客户的密码加密后保存到数据库。
- 保存敏感信息。有些网站需要支付功能，该功能需要用户提供敏感信息，比如信用卡号码。要储存用户的信用卡信息，必须将其加密后才能保存。
- Web 服务安全需要。有些应用需要支持 WS-Security 协议，并采用证书进行消息加密和签名。比如，采用消息安全策略（message security）的 WCF 应用可能需要使用安全证书进行消息加密。
- 数字签名。应用需要对关键信息进行签名，确保信息的完整性和一致性，防止信息在传输或者存储过程中被篡改。

Windows 提供的证书 API 可以从本地计算机（LocalMachine）或当前用户（CurrentUser）的证书库中加载安全证书，也可以从文件系统加载证书。通常，基于安全考虑，安全证书

一般不存放在文件系统中，而是安装到证书库中。

Azure 网站中，用户通过管理门户上传的安全证书，在网站启动时被自动同步到运行用户网站的虚拟机的证书库中。网站应用和 WebJobs 应用可以从个人证书库中加载证书。当网站停止时，这些证书自动从证书库中删除。注意，这只适用于基本模式和标准模式的站点。运行在免费模式和共享模式的站点只能从文件系统中加载证书。

下面的步骤演示了基本模式和标准模式下，Azure 网站中的应用如何使用安全证书。

1. 上传安全证书到 Azure 网站

该步骤与上传 SSL 绑定（HTTPS 传输加密）使用的证书步骤完全相同。可以通过 Microsoft Azure 管理门户网站上传证书，或者通过 Azure PowerShell、X-CLI 命令行上传证书。具体步骤请参考 2.6.2 节。

证书上传后，在管理门户的证书部分，会显示证书的有效期和指纹信息。如图 4-23 所示。

证书

主题	到期日期	指纹
modern.waws.cn	2040/1/1	00239A3E4145CEF79DD4714506168l8627411739
antarestest.trafficmanager.net	2040/1/1	1C9D492CC4BEFD2EE8F4F0F4FC66117111C55371
wzhaodnn.azurewebsites.net	2040/1/1	75DE53FBD9935BE6A2B4ACB8DCE7C029356C3292
obsolete.waws.cn	2040/1/1	C29C916A8C574FC0B14314C9F154249DA0C7856B
legacy.waws.cn	2040/1/1	D08457616B45DDE805BA59E58F1764C32F098D85

上载证书

图 4-23　证书信息

2. 添加安全证书应用设置

Azure 网站默认用户上传的证书用于进行传输加密（HTTPS），而不是用于应用加密数据。因此，默认情况下，用户上传的证书并不会安装到运行用户网站的虚拟机的证书库中。如果需要 Azure 网站将某个证书或多个证书添加到证书库，需要添加一个名为 WEBSITE_LOAD_CERTIFICATES 的应用设置。

添加应用设置的具体步骤请参考 4.3.3.3 小节，具体设置如图 4-24 所示。

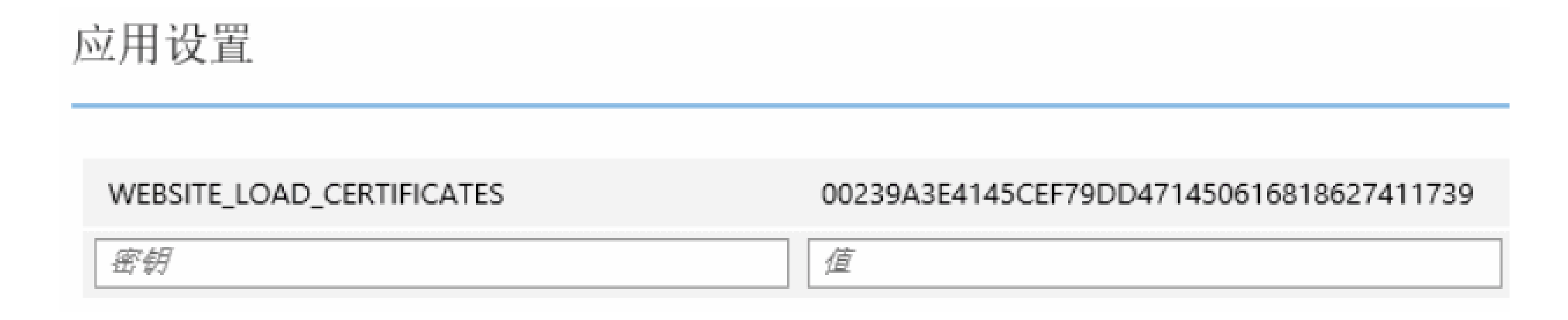


图 4-24　证书应用设置

WEBSITE_LOAD_CERTIFICATES 的值为证书的指纹。如果需要指定多个证书，可以用逗号分隔多个证书的指纹。如果指纹指定通配符（*），则表示所有上传的证书。

3. 在应用中使用证书

将证书添加到证书库后，在应用中使用证书与本地环境完全相同。证书库的位置是当前用户（CurrentUser），证书库的名称为个人证书库（My）。下面的代码演示了 ASP.NET 中如何加载证书：

```
X509Certificate2 cert;
X509Store certStore = new X509Store(StoreName.My, StoreLocation.
CurrentUser);
certStore.Open(OpenFlags.ReadOnly);
X509Certificate2Collection certCollection = certStore.Certificates.Find(
                         X509FindType.FindByThumbprint,
                        // Replace below with your cert's thumbprint
                        "00239A3E4145CEF79DD4714506168l8627411739",
                        false);
// Get the first cert with the thumbprint
if (certCollection.Count > 0)
{
   cert = certCollection[0];
}
certStore.Close();
//use the certificate here
```

对于运行在免费模式和共享模式的站点，只能从文件系统中加载证书，而不能从证书库中加载证书。应用需要将证书部署到 Azure 网站，然后使用.NET Framework 中提供的 API 从文件系统中加载证书。下面的代码从网站根目录的 App_Data 目录下加载密码保护的证书 modern.waws.cn.pfx：

```
X509Certificate2 cert = new
X509Certificate2(@"d:\home\site\wwwroot\app data\modern.waws.cn.pfx",
 "password " );
          //Use certificate
```

4.3.4 Azure 网站 ASP.NET 常见故障查找方法

与运行在本地的 ASP.NET 网站不同，不能完全控制运行在 Microsoft Azure 网站的 ASP.NET 站点。比如，不能远程登录到服务器端进行故障查找。但是，Microsoft Azure 网站仍然提供了很多排错的方法，可以帮助用户快速的定位各种故障。

4.3.4.1 关闭自定义错误

当 ASP.NET 网站服务器端出现异常时，<customErrors>配置项定义了如何查看错误的详细信息。ASP.NET 支持 3 种模式：On、Off 和 RemoteOnly（默认值）。On 表示错误的详细信息不会发送给客户端浏览器。Off 表示用户可以在浏览器上看到服务器端错误的详细

信息。RemoteOnly 表示只有当从服务器本机浏览时才能够查看错误的详细信息。如果 ASP.NET 网站运行在 Microsoft Azure 网站时遇到服务端错误，可以将下面的信息加入 web.config：

```
<system.web>
    <customErrors mode="Off" />
</system.web>
```

如图 4-25 所示，当问题发生时，在浏览器中就会看到错误的具体信息。

Server Error in '/' Application.

Attempted to divide by zero.

Description: An unhandled exception occurred during the execution of the current web request. Please review the stack trace for more informatior

Exception Details: System.DivideByZeroException: Attempted to divide by zero.

Source Error:

```
An unhandled exception was generated during the execution of the current web request. Information regarding t
```

Stack Trace:

```
[DivideByZeroException: Attempted to divide by zero.]
   MyFirstAzureSite.Exception.Page_Load(Object sender, EventArgs e) +92
   System.Web.Util.CalliEventHandlerDelegateProxy.Callback(Object sender, EventArgs e) +51
   System.Web.UI.Control.OnLoad(EventArgs e) +92
   System.Web.UI.Control.LoadRecursive() +54
   System.Web.UI.Page.ProcessRequestMain(Boolean includeStagesBeforeAsyncPoint, Boolean in
```

图 4-25　服务器错误

4.3.4.2　事件日志

Microsoft Azure 网站自动将应用相关的异常保存到 eventlog.xml 文件中，可以通过 FTP 下载 eventlog.xml 文件。该文件位于/LogFiles/目录下。

如图 4-26 所示，在 eventlog.xml 中，会看到具体的错误信息，该信息与图 4-25 所示的信息完全一致。

```
<EventData>
  <Data>3005</Data>
  <Data>An unhandled exception has occurred.</Data>
  <Data>3/18/2014 12:30:23 PM</Data>
  <Data>3/18/2014 12:30:23 PM</Data>
  <Data>14c2f245174746a9b54f395cd73bed30</Data>
  <Data>D:\home\site\wwwroot\</Data><Data>RD00155D38173E</Data>
  <Data>33568</Data><Data>w3wp.exe</Data>
  <Data>IIS APPPOOL\wzhaoFirstAzureSite</Data>
  <Data>DivideByZeroException</Data>
  <Data>Attempted to divide by zero.
at MyFirstAzureSite.Exception.Page_Load(Object sender, EventArgs e)
at System.Web.Util.CalliEventHandlerDelegateProxy.Callback(Object sender,
at System.Web.UI.Control.OnLoad(EventArgs e)
at System.Web.UI.Control.LoadRecursive()
```

图 4-26　事件日志示例

4.3.4.3 详细的错误信息

Microsoft Azure 网站提供了详细的错误信息功能。当 ASP.NET 应用遇到错误或者异常的时候，Microsoft Azure 网站会将错误信息记录下来，供排错使用。

如图 4-27 所示，需要登录到门户管理网站打开该功能。



图 4-27 Azure 网站详细的错误信息

当问题发生后，详细的错误信息保存在/LogFiles/DetailedErrors/目录下，文件命名为 ErrorPage#number.htm。最大错误文件数目为 50 个，超过后新的文件自动覆盖最老的文件。可以通过 FTP 下载详细的错误信息文件。关于如何使用 FTP 访问 Azure 网站的文件，请参考 4.1 节。

例如，如图 4-28 所示，当访问网站时，遇到了服务器端错误。错误信息并没有提供任何关于该错误的原因。

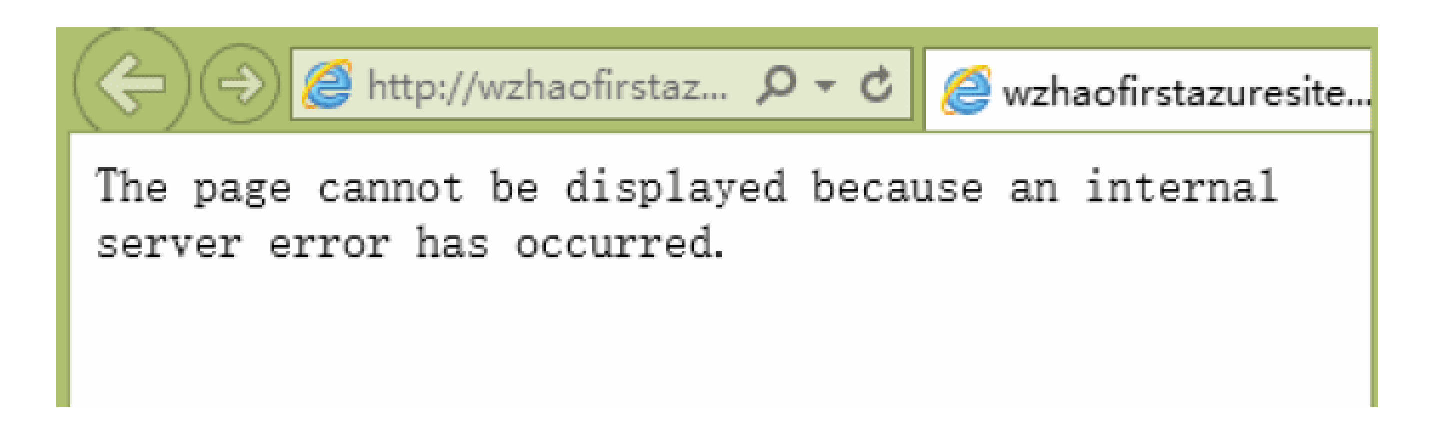


图 4-28 服务器端错误

如图 4-29 所示，可以看到该错误是由于 web.config 中有错误的配置项导致的。

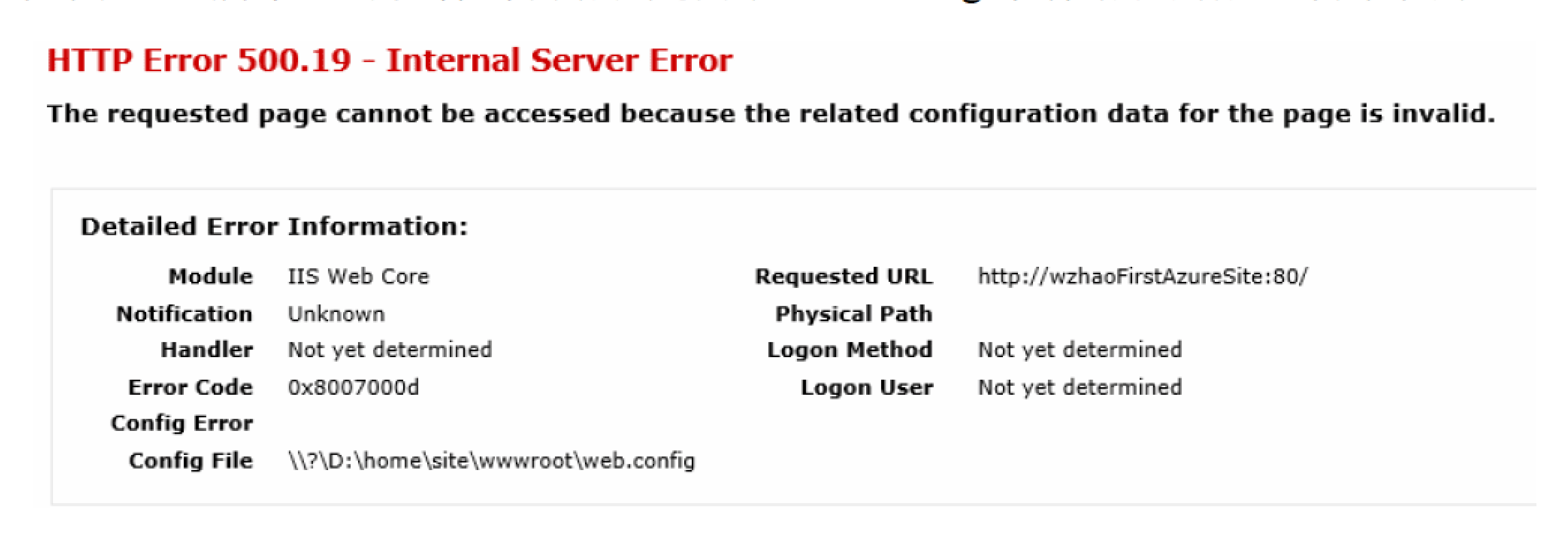


图 4-29 详细错误信息

4.3.4.4　失败请求跟踪

IIS 7 及后续版本提供了基于请求的跟踪功能，该功能提供了一种方法来跟踪整个请求的处理过程。该功能可以帮助我们定位、解决很多问题，比如服务器端 500 错误、性能问题和认证问题等。该功能也被集成在 Microsoft Azure 网站中，但是有如下几个限制：

（1）最大失败请求文件数目为 50，超过后，新的文件自动覆盖旧的文件。

（2）在共享模式和免费模式下，该功能在打开 1 小时后自动关闭。

（3）只跟踪 HTTP 状态码 400～600 的请求。

如图 4-30 所示，可以通过 Azure 管理门户网站的配置页面打开该功能。



图 4-30　失败请求跟踪

如图 4-31 所示，失败请求跟踪清楚地显示 500 服务器错误是由于 ASP.NET 页面中 Page_Load 中的代码错误导致的。

136.	AspNetPagePreInitEnter	
137.	AspNetPagePreInitLeave	
138.	AspNetPageInitEnter	
139.	AspNetPageInitLeave	
140.	AspNetPageLoadEnter	
141.	AspNetHttpHandlerLeave	
142.	AspNetWebEventRaiseStart	Data1="System.Web.Managem
143.	AspNetWebEventRaiseEnd	
144.	MODULE_SET_RESPONSE_ERROR_STATUS Warning	ModuleName="ManagedPipeline HttpStatus="500", HttpReason= completed successfully. (0x0)", ConfigExceptionInfo=""

图 4-31　分析失败请求跟踪日志

4.3.4.5　应用程序跟踪日志

在.NET Framework 中，System.Diagnostics.Trace 类提供了跟踪代码执行的功能，可以帮助用户在不干扰系统运行的情况下隔离并修复问题。Microsoft Azure 网站无缝集成了该功能，并且可以将跟踪日志写入 Microsoft Azure 表存储或者 BLOB 存储中。下面通过前面

创建的 MyFirstAzureSite 工程来演示如何在 Azure 网站中使用应用程序跟踪功能。

（1）加入跟踪代码。在 Visual Studio 中打开 Default.aspx.cs 文件，在 Page_Load 中加入下面的跟踪代码记录函数的进入和离开。修改后的代码如下所示：

```
protected void Page_Load(object sender, EventArgs e)
{
    System.Diagnostics.Trace.TraceInformation("Entered Page_Load");
    this.form1.InnerText = System.DateTime.Now.ToString() + "  ("
    + System.TimeZone.CurrentTimeZone.StandardName + ")";
    System.Diagnostics.Trace.TraceInformation("Left Page_Load");
}
```

（2）将修改后的应用重新部署到 Azure 网站。

（3）打开跟踪功能。登录到 Microsoft Azure 管理门户网站，浏览到网站的配置页面。在“应用程序诊断”部分，将应用程序日志记录功能打开。如图 4-32 所示，Microsoft Azure 网站可以将应用程序跟踪日志存储在文件系统、表存储和 BLOB 存储中。这里以文件系统为例。



图 4-32 打开跟踪功能

对应于 TraceLevel 枚举类型的定义，日志记录级别也分为详细、信息、警告和错误 4 级。关于 TraceLevel 枚举类型的定义，请参见 MSDN 文档：

http://msdn.microsoft.com/zh-cn/library/vstudio/system.diagnostics.tracelevel(v=vs.100).aspx

（4）查看应用程序跟踪日志。打开应用程序日志功能后，再次访问 default.aspx。此时跟踪代码产生的信息将被记录到日志中。如图 4-33 所示，应用程序日志在网站 FTP 的 /LogFiles/Application 目录下。

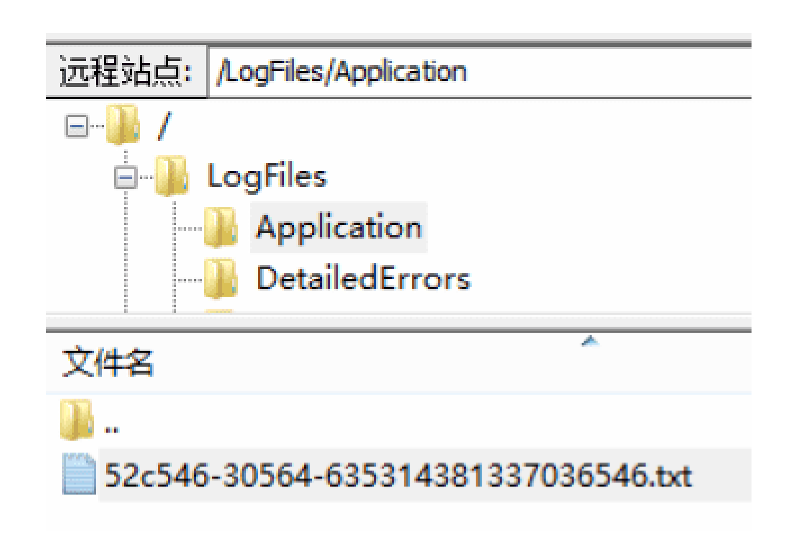


图 4-33 跟踪日志

在日志文件中，可以看到如下信息：

```
2014-07-15T13:42:12  PID[30564] Information Entered Page Load
2014-07-15T13:42:12  PID[30564] Information Left Page_Load
```

（5）使用 PowerShell 和 X-CLI 实时查看日志。在介绍管理自动化的时候提到过 PowerShell 和 CLI 提供了实时查看日志和诊断信息的功能。

运行 Get-AzureWebsiteLog -name wzhaofirstazuresite -Tail 命令，如图 4-34 所示，会实时看到应用程序跟踪输出。

```
PS C:\> select-azuresubscription "Visual Studio Ultimate with MSDN"
PS C:\> Get-AzureWebsiteLog -name wzhaofirstazuresite -Tail
2014-07-15T15:24:56  Welcome, you are now connected to log-streaming service.
2014-07-15T15:25:13  PID[9176] Information Entered Page_Load
2014-07-15T15:25:13  PID[9176] Information Left Page_Load
2014-07-15T15:25:15  PID[9176] Information Entered Page_Load
2014-07-15T15:25:15  PID[9176] Information Left Page_Load
```

图 4-34　PowerShell 查看跟踪信息

运行 azure site log tail wzhaofirstazuresite 命令，如图 4-35 所示，可以查看应用的实时跟踪输出信息。

```
C:\>azure site log tail wzhaofirstazuresite
info:    Executing command site log tail
2014-07-15T15:27:43  Welcome, you are now connected to log-streaming service.
2014-07-15T15:27:49  PID[9176] Information Entered Page_Load
2014-07-15T15:27:49  PID[9176] Information Left Page_Load
2014-07-15T15:27:51  PID[9176] Information Entered Page_Load
2014-07-15T15:27:51  PID[9176] Information Left Page_Load
```

图 4-35　X-CLI 查看跟踪信息

4.3.5　远程调试部署 Azure 网站中的 ASP.NET 站点

Visual Studio 支持从一台计算机上远程调试运行在另一台设备的应用。进行远程调试时，调试机可以是任何支持 Visual Studio 的平台，远程被调试设备可以是 x86、x64 或 ARM 平台。在开发过程中，与跟踪相比，调试功能无疑更直接、更高效。如果问题能够轻松重现，那么调试也是最有效率的手段。

在本章开始介绍了如何通过 Visual Studio 管理 Microsoft Azure 订阅。在 Visual Studio 中集成 Microsoft Azure 的管理功能是 Microsoft Azure SDK 2.2 版本中添加的重大改进之一。该功能使得开发人员能够在 Visual Studio 中直接开发、测试和管理 Microsoft Azure 中部署的资源。其中，远程调试部署在 Microsoft Azure 网站中的 ASP.NET 站点无疑是最让开发人员兴奋的功能。下面具体演示如何使用 Visual Studio 2013 远程调试部署在 Microsoft Azure 网站中的 ASP.NET 站点。

1. 部署 Debug 版本的应用到 Microsoft Azure 网站

为了获取更好的调试支持，建议部署 Debug 版本的 ASP.NET 应用到 Microsoft Azure

网站。如图 4-36 所示，要发布 Debug 版本的 ASP.NET 应用，需要在发布设置对话框中的“配置”下拉框中选择 Debug。



图 4-36 发布 Debug 版本应用

2．打开远程调试功能

登录到 Microsoft Azure 管理门户网站，在网站的配置页面中，如图 4-37 所示，打开远程调试功能。



图 4-37 打开远程调试

注意：

（1）远程调试功能在启用 48 小时后会自动关闭。之后，如果需要继续调试，需要再次打开远程调试功能。

（2）目前远程调试功能支持 Visual Studio 2012 和 Visual Studio 2013 两个版本，两个版本之间互相不兼容。如果使用的是 Visual Studio 2013，那么需要在管理门户网站选择 2013。

（3）远程调试功能使用 NTLM v2 进行身份认证，Visual Studio 在连接远程调试服务时，自动使用 NTLM 进行认证。所以，正常情况下不需要输入用户名和密码。如果 Visual Studio 弹出对话框要求输入用户名和密码，表示 NTLM 认证失败了，需要检查本地的 NTLM 相关的设置。

3．在 Visual Studio 中设置断点

与本地调试相同，可以在 Visual Studio 中设置断点。如图 4-38 所示，在 Default.aspx.cs 文件中，在 Page_Load 方法中设置一个断点。

```
namespace MyFirstAzureSite
{
    1 个引用
    public partial class Default : System.Web.UI.Page
    {
        0 个引用
        protected void Page_Load(object sender, EventArgs e)
        {
            System.Diagnostics.Trace.TraceInformation("Entered Page_Load");
            this.Title = System.DateTime.Now.ToString() + "   (" + System.Ti
            this.form1.InnerText = System.DateTime.Now.ToString() + "   (" +
            System.Diagnostics.Trace.TraceInformation("Left Page_Load");
        }
    }
}
```

图 4-38　设置断点

4. 开始远程调试

如图 4-39 所示，在 Visual Studio 中的“服务器资源管理器”中，右击要调试的网站，选择“附加调试器”。此时 Visual Studio 连接到远程调试服务，并自动启动一个 IE 浏览器打开要调试的网站。此时可以在浏览器中重现该网站的问题。

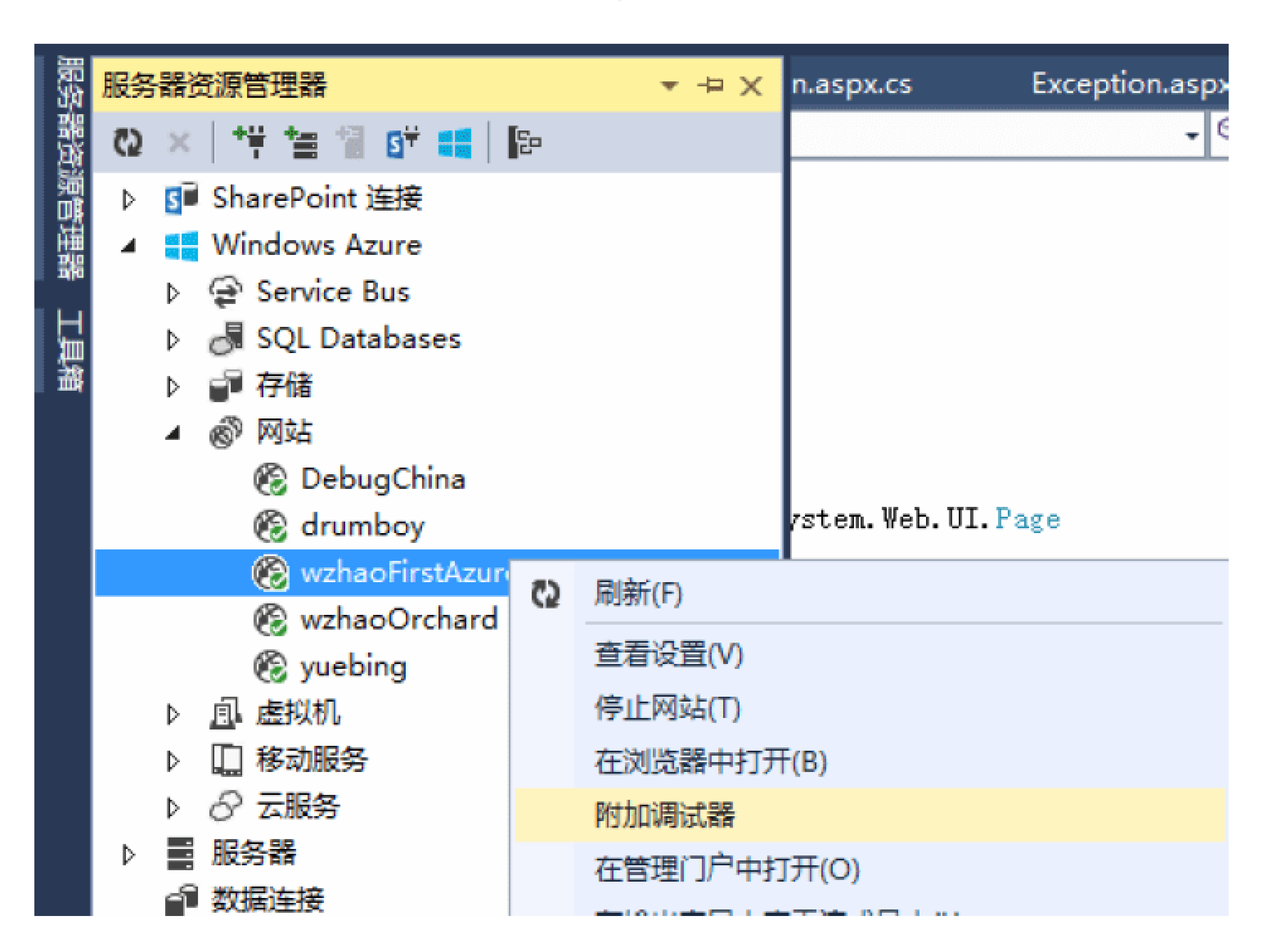


图 4-39　附加调试器

注意：如果没有在 Microsoft Azure 管理门户网站中打开远程调试功能，Visual Studio 此时会自动调用 Azure 网站的 REST API 打开远程调试功能。

5. 在 Visual Studio 中实时调试

当设置断点的代码被执行时，调试器自动中断程序的运行，进入调试状态。此时，如同本地调试一样，可以在 Visual Studio 中交互式地调试部署在 Azure 中的网站，如图 4-40 所示。

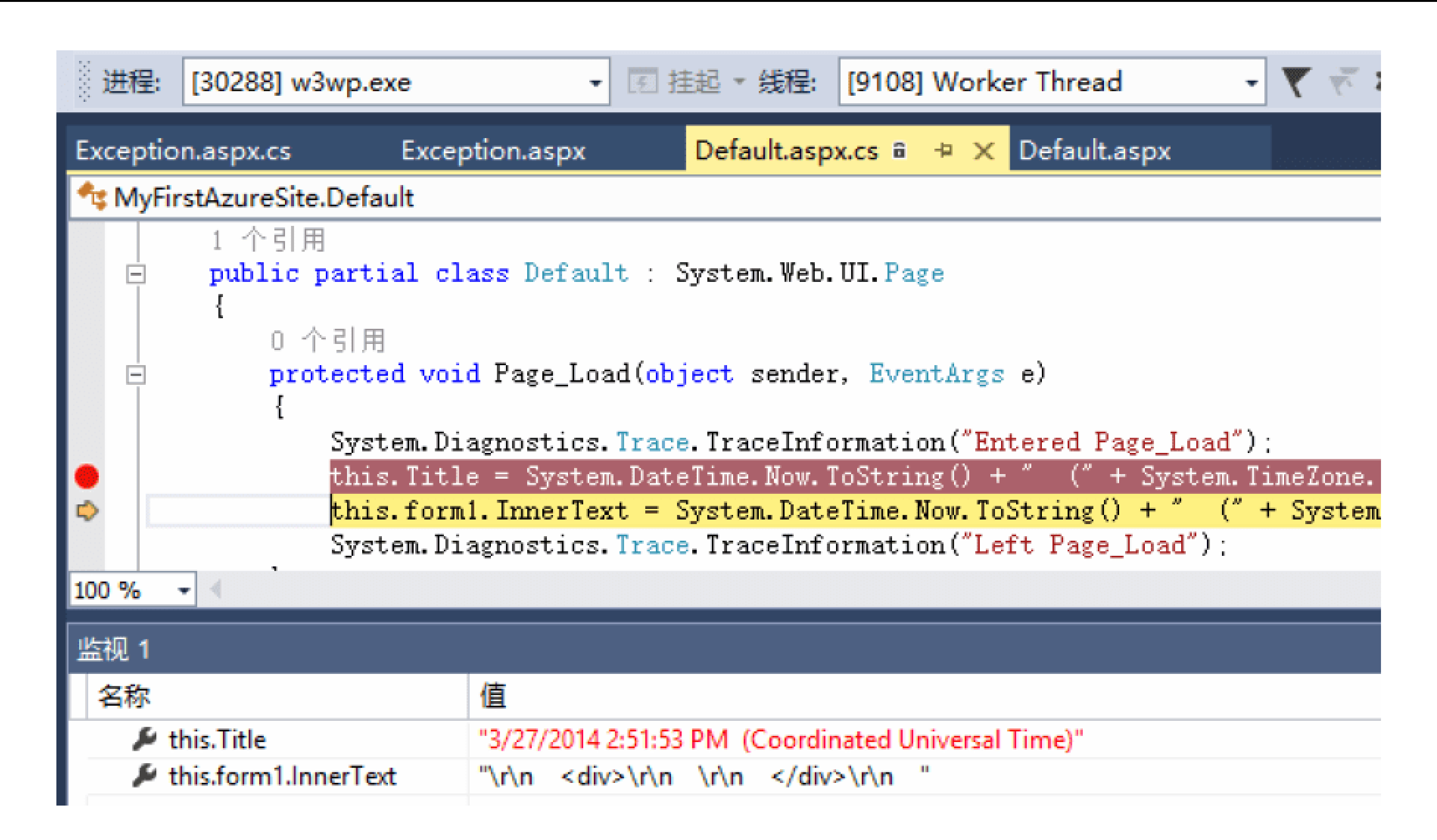


图 4-40　在 Visual Studio 中实时调试

正如上面所见，远程调试功能提供了与本地调试一致的调试功能和开发体验。尤其是在开发过程中，远程调试极大地提高了开发人员的排错效率。

4.3.5.1　Azure ASP.NET 网站排错方法小结

虽然与 Microsoft Azure 云服务或者运行在企业内部的 ASP.NET 网站不同，当网站运行出现问题后，不能直接远程登录到运行网站的服务器进行排错。但是 Microsoft Azure 网站依然提供了极佳的排错/调试功能和体验。

当网站运行遇到异常时，首先需要检查 eventlog.xml 文件。如果 ASP.NET 网站遇到了异常情况，该文件包含与 ASP.NET 应用相关的异常信息，比如异常发生的时间、异常的具体解释、调用堆栈等。

详细的错误信息也会包含更多的信息，该信息对于配置文件错误尤其有帮助。

失败请求跟踪（FREB）对于定位哪个模块导致异常非常有帮助。

如果 ASP.NET 网站本身包含应用跟踪功能，那么应用程序跟踪日志可以记录应用发生的具体事件，可以协助分析问题发生的原因。

最后，如果问题可以比较方便地重现，那么远程调试功能可以帮助开发人员快速定位问题。

4.4　Azure 网站上的 PHP 开发

Microsoft Azure 网站原生无缝支持 PHP。在 Microsoft Azure 网站中，创建一个 PHP 网站与创建任何其他类型的网站没有任何不同。

在本节中，首先讨论 Azure 网站上 PHP 的架构，然后详细讨论如何配置 PHP 运行时、PHP 扩展（Extension）以及 PHP 网站问题排错。

4.4.1　Azure 网站中 PHP 架构

图 4-41 描述了 Windows Azure 网站上 PHP 的架构。该架构与本地 IIS 上的 PHP 配置并无不同。

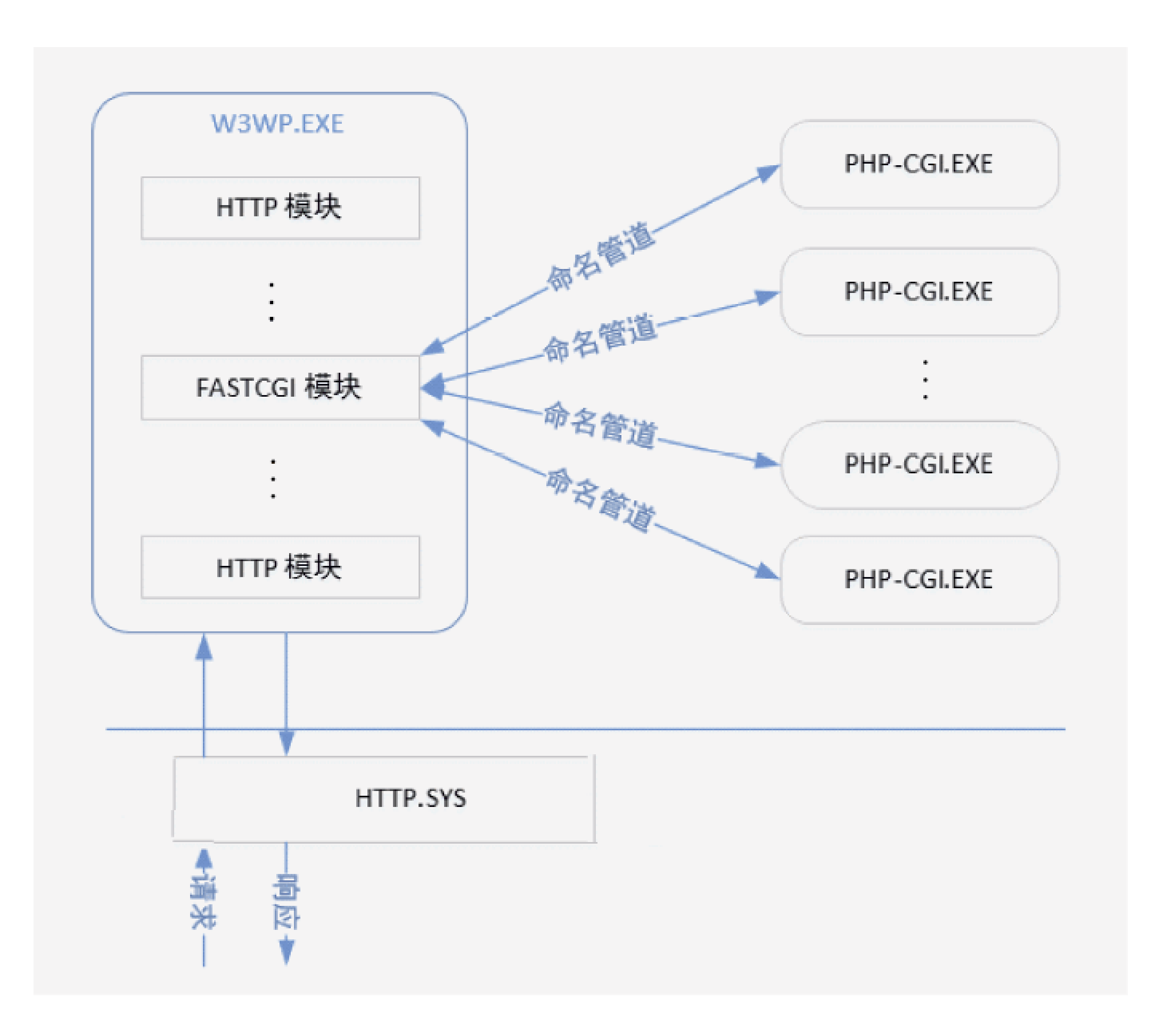


图 4-41　Azure 网站中 PHP 架构

Azure 网站中，PHP 请求的处理流程如下：

（1）客户端 HTTP 请求到达 HTTP.SYS（HTTP.SYS 为负责处理 HTTP 请求的内核模块）。

（2）HTTP 请求被转发到网站的工作进程 W3WP.EXE。

（3）IIS 将 PHP 请求交给 FASTCGI 模块负责处理。

（4）根据具体情况，FASTCGI 模块启动一个新的 PHP-CGI.EXE，然后将请求转发到新的 PHP-CGI.EXE，或者将请求转发到一个现有的 PHP-CGI.EXE。FASTCGI 与 PHP-CGI.EXE 之间的通信采用命名管道。

（5）PHP-CGI.EXE 执行 PHP 页面，然后将结果返回给 FASTCGI 模块。

（6）最终，响应通过 HTTP.SYS 发回客户端。

在 Azure 网站上部署 PHP 时，需要注意下面的事项：

（1）Azure 网站默认启用 PHP，如果不需要 PHP，可以通过管理门户网站关闭 PHP 功能。

（2）Azure 网站不允许客户修改 PHP 的系统级别设置。

（3）Azure 网站上安装的是非线程安全 PHP 版本（NTS）。

（4）Azure 网站目前支持 PHP 5.3、5.4、5.5 版本，后面会讨论如何配置其他版本。

（5）Azure 网站目前只支持 32 位 PHP。在管理门户网站将网站设置为 64 位模式，只是将 W3WP.EXE 设置为 64 位。PHP-CGI.exe 仍然是 32 位。

（6）PHP-CGI.EXE 的实例数目由 FASTCGI 模块根据负载和机器资源情况自动决定。

4.4.2　配置 Azure 网站上的 PHP

4.4.2.1　指定 PHP 版本

登录到 Microsoft Azure 管理门户网站，在网站的配置页面，在常规配置项下，可以看到 PHP 版本配置。如图 4-42 所示，在默认情况下，所有网站默认启用 PHP。目前 Microsoft Azure 网站支持 PHP 5.3、5.4 和 5.5，默认版本为 PHP 5.4。



图 4-42　PHP 版本

1．PHP 配置分类

PHP 配置项分为 4 类，表 4-1 描述了这 4 类配置选项的具体作用域。

表 4-1　PHP 配置项分类

类　型	作 用 域
PHP_INI_USER	可在用户脚本（例如通过 ini_set()）或者.user.ini 文件中设定
PHP_INI_PERDIR	可在 php.ini 和.htaccess 中设定
PHP_INI_SYSTEM	可在 php.ini 中设定
PHP_INI_ALL	可在任何地方设定

关于 PHP 配置选项分类以及作用域，请参考下面的文章：

http://php.net/manual/en/configuration.changes.modes.php

对于运行在 Microsoft Azure 的 PHP 网站而言，可以修改非系统级别的 PHP 配置。基于安全考虑，不能修改系统级别的配置选项。其中系统级别配置是指类型为 PHP_INI_SYSTEM 的配置项，这些配置项只能配置在 php.ini 文件中。修改系统级别的配置会影响所有的用户，因此 Azure 网站不允许修改系统级别的 PHP 配置选项。

下面的 PHP 文档列出了所有的 PHP 配置项：

http://www.php.net/manual/en/ini.list.php

2．使用.user.ini 文件修改 PHP 配置选项

很多情况下，应用需要修改非系统级别的默认配置。比如，默认情况下，Azure 网站

中的 PHP 运行环境最大允许上传 8MB 的文件。如果网站允许客户上传更大的文件，比如视频文件，需要修改 upload_max_filesize 和 post_max_size 配置。默认情况下，PHP 的最长执行时间为 30s。而在 Microsoft Azure 网站中，默认为 300s。如果希望 PHP 脚本执行更长时间，那么需要修改 max_execution_time 设置。下面的步骤具体演示了在 Microsoft Azure 网站中如何修改非系统级别的配置选项。

（1）创建一个.user.ini 文件，该文件包含要修改的配置项。下面的配置将最大上传文件设置为 16MB，将 PHP 执行超时时间设置为 120s。

```
; allow upload files up to 16MB
upload max filesize=16MB
; post max size > upload max filesize
Post_max_size=16MB
; allow PHP script run as long as 2 minutes
max_execution_time=120
```

（2）将该文件部署到网站的根目录下。

（3）登录到 Microsoft Azure 管理门户网站，重新启动网站使修改立即生效。在 Microsoft Azure 网站的 PHP 设置中，user_ini.cache_ttl 被设置为 5 分钟。该选项控制 PHP 保存配置的缓存时间，意味着如果不重启网站，那么修改可能需要 5 分钟才能生效。

（4）可以通过 phpinfo()函数来验证修改。比如创建一个 phpinfo.php 包含如下内容，并将该文件上传到网站的根目录：

```
<?php phpinfo() ?>
```

（5）打开浏览器访问 phpinfo.php 文件，在 Configuration 部分会看到设置覆盖了默认设置。图 4-43 显示了 upload_max_filesize 的修改结果。16M 为网站设置，8M 为全局的默认设置。

unserialize_callback_func	*no value*	*no value*
upload_max_filesize	16M	8M
upload_tmp_dir	C:\DWASFiles\Sites\php-mysql\Temp	C:\DWASFiles\Sites\mysql\Temp

图 4-43　修改后的 upload_max_filesize

3．使用 ini_set 函数修改 PHP 配置选项

.user.ini 文件的设置应用于整个 PHP 站点。如果只希望针对某个 PHP 页面修改配置，那么可以选择使用 ini_set()函数。Ini_set()函数修改的配置项只应用于本页面中，在本页面执行结束后恢复默认设置，不影响其他页面。Ini_set()函数的具体函数定义和示例请参考 PHP 文档：

http://www.php.net/manual/en/function.ini-set.php

下面通过一个简单的场景来解释如何使用 ini_set 函数来修改 PHP 配置项。

在 Azure 网站中部署了一个页面（error.php），但是该页面总是返回空白页面。如果使

用 IE 调试功能，发现该页面返回 500 错误。为了获取更多的页面错误信息，编写一个简单页面如下：

```
<?php
  ini set('display errors', 'stdout');
  include("error.php");
?>
```

将该页面部署到 Azure 网站，访问该页面后，可以看到如图 4-44 所示的信息，非常简单直接地表明 error.php 的第二行有语法错误。

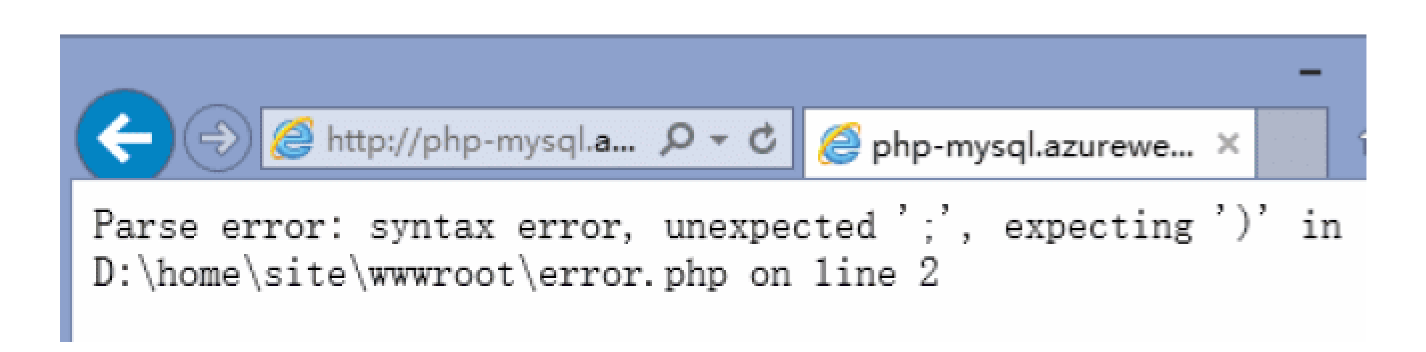


图 4-44　PHP 页面显示详细错误信息

4.4.2.2　使用自定义 PHP 版本

Microsoft Azure 网站提供了 3 种版本的 PHP，并允许修改非系统级别的配置。某些特殊的情况下，可能仍然无法满足要求，比如：

- 您希望使用特定版本的 PHP。

有些 PHP 应用可能只能支持特定版本的 PHP，比如 PHP 5.2，而 Azure 网站并不支持 PHP 5.2。或者用户希望使用最新的 beta 版本的 PHP。

- 及时的补丁及安全更新。

Microsoft Azure 网站的 PHP 版本不是实时更新。比如应用遇到了 PHP 的缺陷，在最新的版本中已经修复。但是，Microsoft Azure 网站可能需要 2～3 周的时间进行验证并更新到最新版本。

- 关闭不需要的 PHP 扩展模块。
- 使用 64 位 PHP。

当遇到上面的情况时，可以使用自定义的 PHP 版本。下面的具体步骤演示了如何在 Microsoft Azure 网站中配置并使用 PHP 5.6 Alpha 版本。

（1）下载 PHP 5.6 Alpha 的非线程安全版本。

Microsoft Azure 网站只支持非线程安全的 PHP 版本，可以从下面的地址下载最新版本的 PHP：

http://windows.php.net/download/

如果需要老的 PHP 版本，可以在存档中找到：

http://windows.php.net/downloads/releases/archives/

（2）配置并上传 PHP 到 Microsoft Azure 网站。

根据应用需要修改 PHP 5.6 Alpha 版本的 php.ini 文件，包括启用或者禁用 PHP 扩展模块。请注意，任何系统级别配置都会被 Microsoft Azure 网站忽略。

然后，将 PHP 5.6 Alpha 版本的所有文件上传到网站，比如 bin/php56 目录下。

（3）配置处理程序映射。

登录到 Microsoft Azure 管理门户网站，在网站的配置页面，定位到“处理程序映射”部分。如图 4-45 所示，添加一个*.PHP 扩展，将其映射到前面上传的 PHP 5.6 Alpha 版本。网站的根目录为 d:\home\site\wwwroot，因此 PHP 5.6 脚本处理器的路径为 d:\home\site\wwwroot\bin\php56\php-cgi.exe。

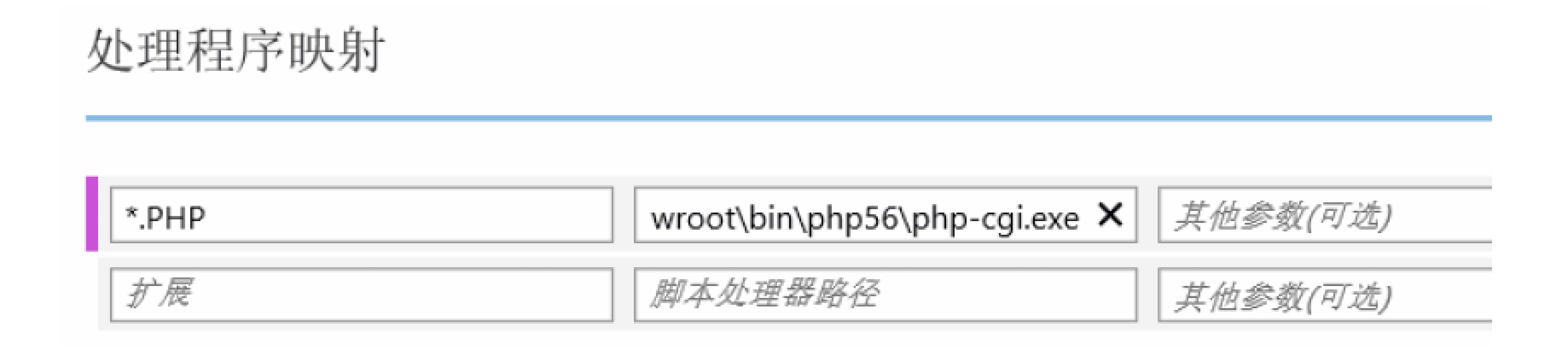


图 4-45　配置自定义 PHP 版本

（4）单击页面底部命令栏的“保存”按钮。

（5）现在，可以通过 phpinfo()函数来验证修改。比如创建一个 phpinfo.php 包含如下内容，并将该文件上传到网站根目录：

```
<?php phpinfo() ?>
```

（6）浏览该文件，如图 4-46 所示， PHP 版本已经变成了 PHP 5.6 Alpha 版本。

PHP Version 5.6.0alpha3

System	Windows NT RD00155D3819
Build Date	Feb 27 2014 16:31:50
Compiler	MSVC11 (Visual C++ 2012)
Architecture	x86

图 4-46　自定义 PHP 版本

4.4.3　配置 PHP 扩展模块

Microsoft Azure 网站允许客户启用自定义 PHP 扩展模块。Microsoft Azure 网站的 PHP 默认启用了很多常用 PHP 扩展模块。具体的信息可以参考 phpinfo()输出结果的 EXTENSION 部分。很多情况下，应用需要自定义的扩展模块，比如使用 PHP Mongo DB 扩展模块连接 Mongo DB，或者需要启用 XDebug 调试扩展模块。Microsoft Azure 网站提供了一个非常方便的方式来启用自定义扩展。Microsoft Azure 网站支持两种 PHP 扩展：PHP 扩展模块和 ZEND 扩展模块。下面分别介绍如何启用这两种扩展模块。

4.4.3.1　启用 PHP 扩展模块

下面以 Mongo DB PHP 扩展模块和 APC（Alternative PHP Cache）为例演示如何启用 PHP 扩展模块。示例网站为 32 位 PHP 5.4。

（1）下载适用于 PHP 5.4 的非线程安全（NTS）Mongo DB PHP 扩展模块：

http://pecl.php.net/package/mongo

（2）下载适用于 PHP 5.4 的非线程安全（NTS）APC 扩展模块：

http://pecl.php.net/package/APC/3.1.13/windows

（3）将 php_apc.dll 和 php_mongo.dll 上传至网站的/wwwroot/bin 目录下。

（4）配置 PHP 扩展模块。

① 登录到 Microsoft Azure 管理门户网站，选择要配置的网站，在网站配置页面，定位到“应用设置”部分。

② 如图 4-47 所示，在左侧输入 PHP_EXTENSIONS，在右侧输入 Mongo DB 和 APC 扩展模块的路径，多个文件中间用逗号隔开。在本例中，路径为 d:\home\site\wwwroot\bin\php_mongo.dll,d:\home\site\wwwroot\bin\php_apc.dll。

应用设置

WEBSITE_NODE_DEFAULT_VERSION	0.10.21
PHP_EXTENSIONS	d:\home\site\wwwroot\bin\php_mongo.dll,d:\home'

图 4-47 PHP_EXTENSIONS 应用设置

（5）可以通过 phpinfo()函数来验证修改。比如创建一个 phpinfo.php 包含如下内容，并将该文件上传到您的网站根目录：

```
<?php phpinfo() ?>
```

（6）如图 4-48 所示，Mongo DB 和 APC 扩展已经被启用。

apc

APC Support	enabled
Version	3.1.13
APC Debugging	Enabled
MMAP Support	Disabled
Locking type	Windows Slim RWLOCK (native)
Serialization Support	php
Revision	$Revision: 327136 $
Build Date	Oct 21 2013 13:24:25

mongo

MongoDB Support	enabled
Version	1.5.0RC1
Streams Support	enabled
SSL Support	disabled
Supported Authentication Mechanisms	
MONGODB-CR (default)	enabled
MONGODB-X509	enabled
GSSAPI (Kerberos)	disabled
PLAIN	disabled

图 4-48 启用的 PHP 扩展模块

4.4.3.2　启用 ZEND 扩展模块

下面以 xdebug 扩展模块为例，演示如何启用 ZEND 扩展模块。

（1）下载非线程安全的 PHP XDebug 扩展模块。

可以从下面的网站下载相应版本的 PHP xdebug 扩展模块（非线程安全版本）。比如，如果站点使用 32 位 PHP 5.5，那么需要下载适用于 PHP 5.5 的 xdebug 2.2.4 x86 版本，对应的文件名称为 php_xdebug-2.2.4-5.5-vc11-nts.dll：

http://xdebug.org/download.php

（2）使用 FTP 将该 DLL 上传至网站，比如/site/wwwroot/bin/xdebug 目录，它对应的物理路径为 d:\home\site\wwwroot\bin\xdebug。

（3）配置 ZEND 扩展模块。

① 登录到 Microsoft Azure 管理门户网站，选择要配置的网站，在网站配置页面，定位到“应用设置”部分。

② 如图 4-49 所示，在左侧输入 PHP_ZENDEXTENSIONS，右侧输入模块对应的文件的路径。

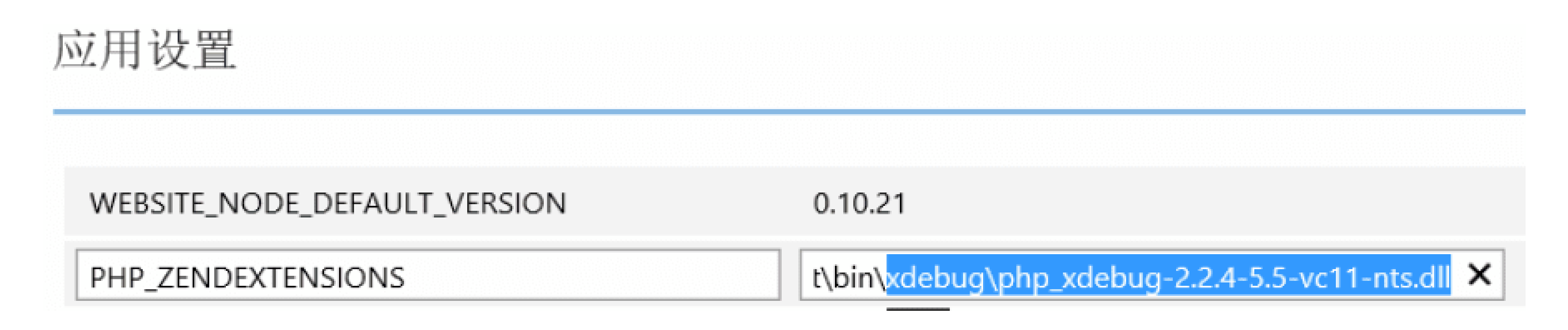


图 4-49　配置 xdebug 扩展模块

（4）可以通过 phpinfo()函数来验证修改。比如创建一个 phpinfo.php 包含如下内容，并将该文件上传到网站的根目录：

```
<?php phpinfo() ?>
```

（5）如图 4-50 所示，xdebug 已经被启用。

xdebug

xdebug support	enabled
Version	2.2.4
IDE Key	RD00155D3819B4$

图 4-50　启用 xdebug 扩展模块

（6）如果需要配置 xdebug 扩展模块，比如启用 xdebug 的 Profiling 功能。需要创建一个包含如下内容用的.user.ini 文件，并通过 FTP 上传到/site/wwwroot 目录下：

```
zend_extension = "d:\home\site\wwwroot\bin\php_xdebug-2.2.4-5.5-vc11-nts.dll"
xdebug.profiler_enable=1
xdebug.profiler_output_dir="D:\home\LogFiles"
```

4.4.3.3 加载 PHP 扩展模块注意事项

（1）指定正确的扩展模块类型。如果是 PHP 扩展模块，在管理门户网站需要指定 PHP_EXTENSIONS，如果是 ZEND 扩展模块，则需要指定 PHP_ZENDEXTENSIONS。

（2）配置正确的版本。比如，网站使用的是 PHP 5.4，则在配置 PHP 扩展模块时，要确保该模块支持 PHP 5.4。如果网站使用的是 64 位 PHP，需要配置 64 位的扩展模块。

（3）如果 PHP 扩展模块依赖于其他的模块，请同时上传这些模块。时刻牢记 Azure 网站环境只是一个单纯的 Windows 系统，不包含任何其他的模块。

（4）Microsoft Azure 网站支持同时启用多个扩展模块。要启用多个扩展模块，同时指定多个扩展模块文件名，用逗号分隔。比如：

```
Bin\PHP_XDebug.dll,bin\PHP_Mongodb.dll
```

4.4.4 PHP 网站排错

4.4.4.1 php_errors.log

Microsoft Azure 网站自动记录 PHP 站点的错误日志，包括 PHP 页面运行过程中产生的异常，也包括扩展模块相关的异常信息。这些错误日志记录在/LogFiles/php_errors.log 文件中，可以通过 FTP 下载该文件。

下面是 php_errors.log 中的一些实例：

```
    [09-Jun-2014 14:04:24 America/Los Angeles] PHP Notice:  Undefined index: countrycode in D:\home\site\wwwroot\cp\messenger\longcodes.php on line 127
    [09-Jun-2014 14:04:25 America/Los Angeles] PHP Warning: file get contents(http://rest.nexmo.com/number/search?api key=5b5fd588&api secret=a1b9abba&features=SMS&country=): failed to open stream: HTTP request failed! HTTP/1.1 420 in D:\home\site\wwwroot\cp\messenger\longcodes.php on line 128
    [09-Jun-2014 14:04:25 America/Los_Angeles] PHP Warning:  Invalid argument supplied for foreach() in D:\home\site\wwwroot\cp\messenger\longcodes.php on line 142
```

4.4.4.2 使用 error_log 函数

与 ASP.NET 中的 trace 功能类似，可以在 PHP 代码中使用 error_log 函数将应用程序的信息记录到 php_errors.log 中。下面的例子记录了进入和离开函数的时间：

```
function DoYourWork()
{
// Entered DoYourWork
error log ( "Enter DoYourWork " . date('h:i:s') );
… function body ….
```

```
//Leaving DoYourWork
error log ( "Leaving DoYourWork " . date('h:i:s') );
}
```

在 php_errors.log 中，会看到类似下面的信息：

```
[02-Apr-2014 00:42:05 America/Los_Angeles] Enter DoYourWork 12:42:05
[02-Apr-2014 00:43:16 America/Los_Angeles] Leaving DoYourWork 12:43:16
```

4.4.4.3　使用 xdebug 模块

xdebug 是一个开源 PHP 程序调试器，可以用来跟踪、调试和分析 PHP 程序的运行状况。目前，Microsoft Azure 网站不支持远程调试 PHP 站点。但是，如前面所示，仍然可以使用 xdebug 扩展模块排错 PHP 网站问题。

下面具体介绍如何使用 xdebug 的性能分析功能查找 PHP 网站的性能瓶颈。

（1）如 4.4.3 节所述，配置 xdebug 并使用.user.ini 文件打开 profiler 功能。启用后，如图 4-51 所示，在/LogFiles 目录下会生成 cachegrind.out 文件。

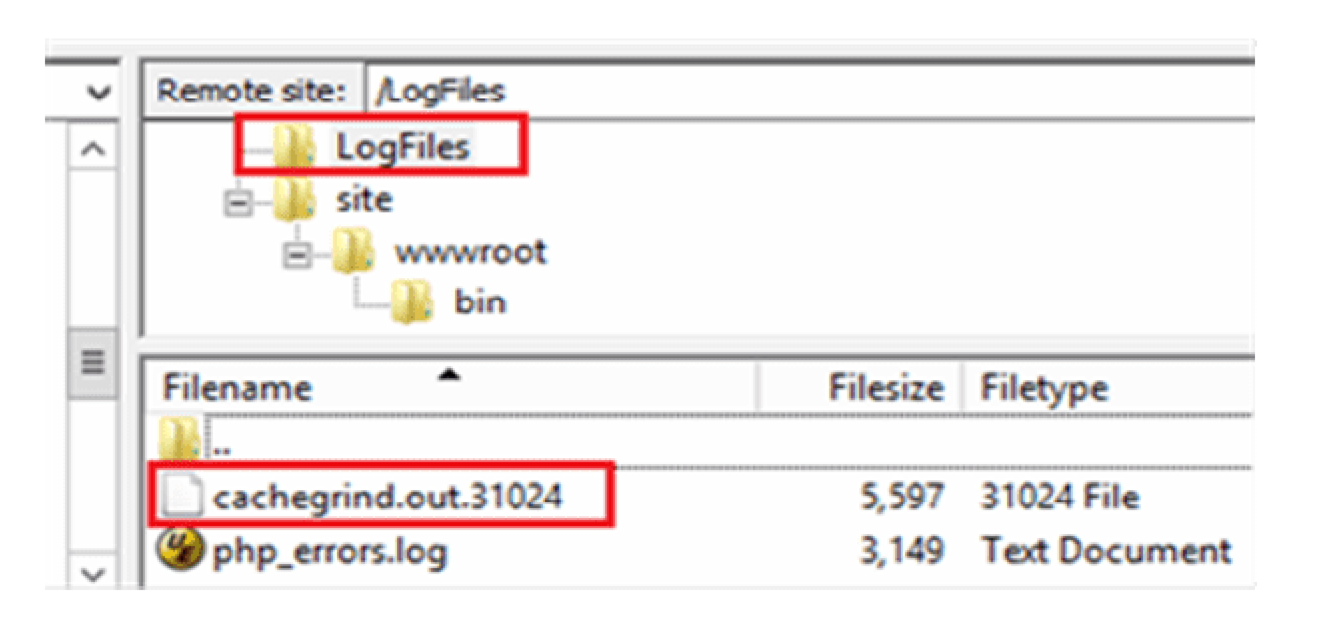


图 4-51　xdebug 生成的性能日志文件

（2）使用 WinCacheGrind 工具可以分析 xdebug 的性能文件，WinCacheGrind 工具可以从 SourceForge 网站下载：

http://sourceforge.net/projects/wincachegrind/

如图 4-52 所示，可以非常直观地看到 SlowFunction 使用了 74.180s 的时间。

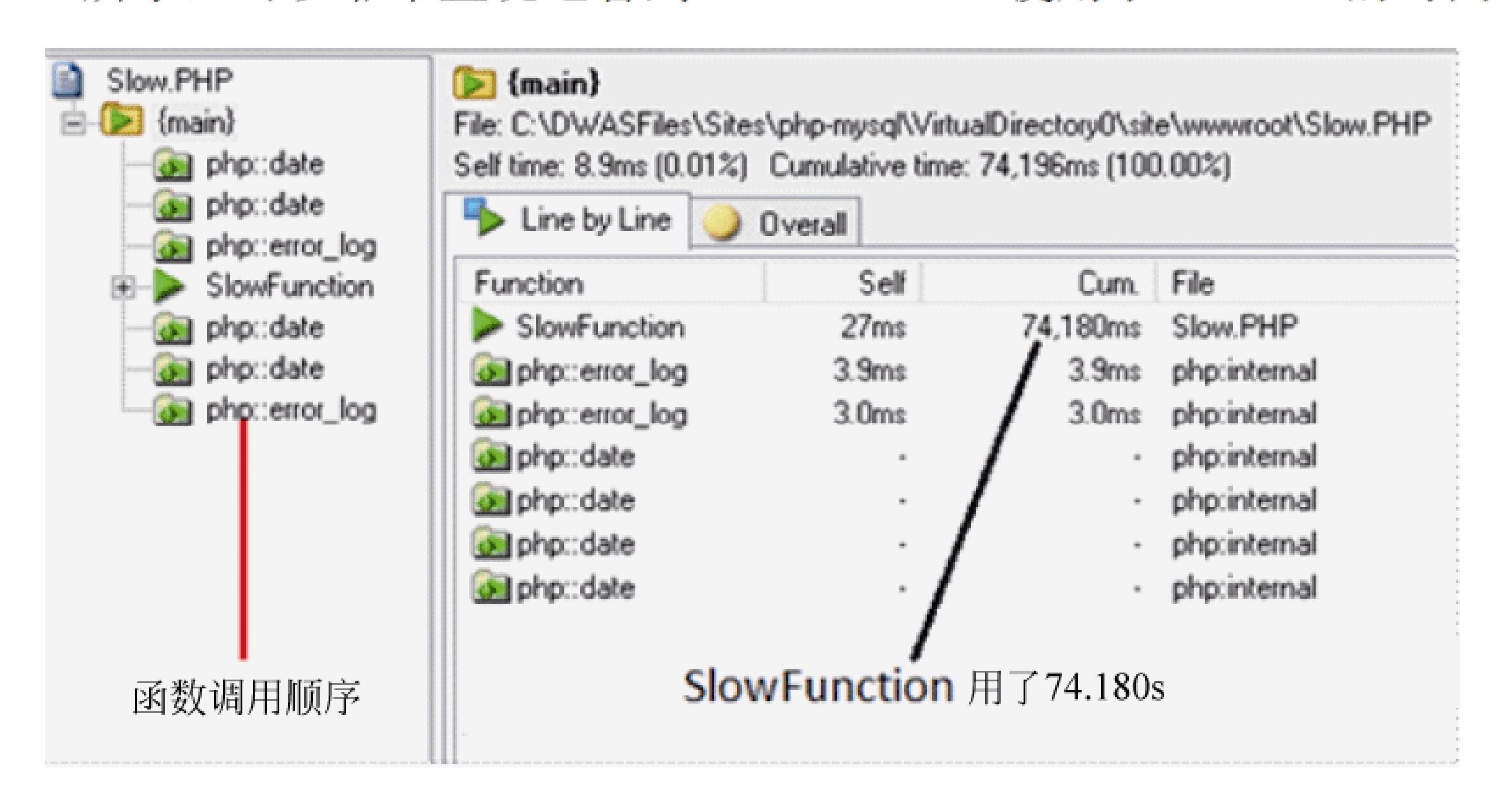


图 4-52　分析 xdebug 性能日志文件

在图 4-52 所示的界面中，双击 SlowFunction，可以看到图 4-53 所示的具体信息。SlowFunction 被调用了 mySleep 10 次，每次花费 5～10s。

Function	Self	Cum.	File
mySleep	2.3ms	10,013ms	Slow.PHP
mySleep	2.4ms	9,016ms	Slow.PHP
mySleep	2.4ms	9,010ms	Slow.PHP
mySleep	5.1ms	8,019ms	Slow.PHP
mySleep	8.0ms	8,014ms	Slow.PHP
mySleep	3.8ms	7,012ms	Slow.PHP
mySleep	13ms	6,023ms	Slow.PHP
mySleep	7.5ms	6,018ms	Slow.PHP
mySleep	4.3ms	6,016ms	Slow.PHP
mySleep	2.6ms	5,013ms	Slow.PHP

图 4-53 查看函数调用耗时

4.4.4.4 按需启动 xdebug profiler

在前面的例子中，使用了 xdebug.profiler_enabled=1 来启用 xdebug 的 profiler 功能。该设置应用到整个网站所有的 PHP 页面，会导致整个网站的性能进一步变慢，在生产环境中可能会导致严重的客户体验问题。在这种情况下，可以使用触发器来启用 xdebug 的 profiler 功能。当 xdebug.profiler_enable_trigger 被设置为 1，正常的页面访问不受影响。需要通过使用 XDEBUG_PROFILE 的 GET/ POST 参数，或者设置一个名为 XDEBUG_PROFILE 的 cookie 来触发 xdebug 的 profiler 功能。通常，使用 xdebug_enable_trigger 的时候，需要设置 xdebug.profiler_enable 为 0。如果需要通过触发器来启用 xdebug 的 profiler 功能，要在网站的根目录下创建一个.user.ini 文件，包含下面的内容：

```
zend extension = ".\bin\php xdebug-2.2.3-5.5-vc11-nts-x86 64.dll"
xdebug.profiler enable=0
xdebug.profiler output dir="D:\home\Logfiles"
xdebug.profiler_enable_trigger=1
```

比如，有一个页面 slow.php 响应缓慢。如果希望查找缓慢的原因，此时可以通过访问 HTTP://mysite.azurewebsites.net/slow.php?XDEBUG_PROFILE=1 来启用 xdebug 的 profiler 功能。如果需要 POST 请求来重现问题，可以使用 Fillder 等工具来构建 HTTP 请求，或者可以使用 xdebug 的浏览器扩展。可以在下面的网站下载 xdebug 的浏览器扩展：

http://xdebug.org/docs/remote#firefox-ext

4.5 Azure 网站上的 Node.js

Node.js 是一个基于 Google V8 引擎的，采用事件驱动的服务器端的 JavaScript 运行平台。它采用事件驱动的非阻塞 I/O 模式，以单线程方式运行。它的目标是提供一种构建可伸缩网络程序的简单方法。

关于 Node.js 的更多信息，请访问 Node.js 的官方网站：

http://nodejs.org

Node.js 自 2009 年诞生以来发展迅速，目前已经得到广泛的应用和关注。Azure 网站无缝支持 Node.js。在本节中，讨论 Node.js 在 Azure 网站上的架构、配置和排错手段。

4.5.1　Azure 网站上 Node.js 的架构

图 4-54 描述了 Azure 网站上的 Node.js 的架构。该架构的核心是 IISNode 模块。IISNode 主要有两个功能：

（1）进程管理。IISNode 负责 Node.EXE 进程的管理。比如创建、回收和监控 Node.EXE 进程等。

（2）请求转发。将 Node.js 请求转发至 Node.EXE 进行处理，并将处理结果返回给客户端。

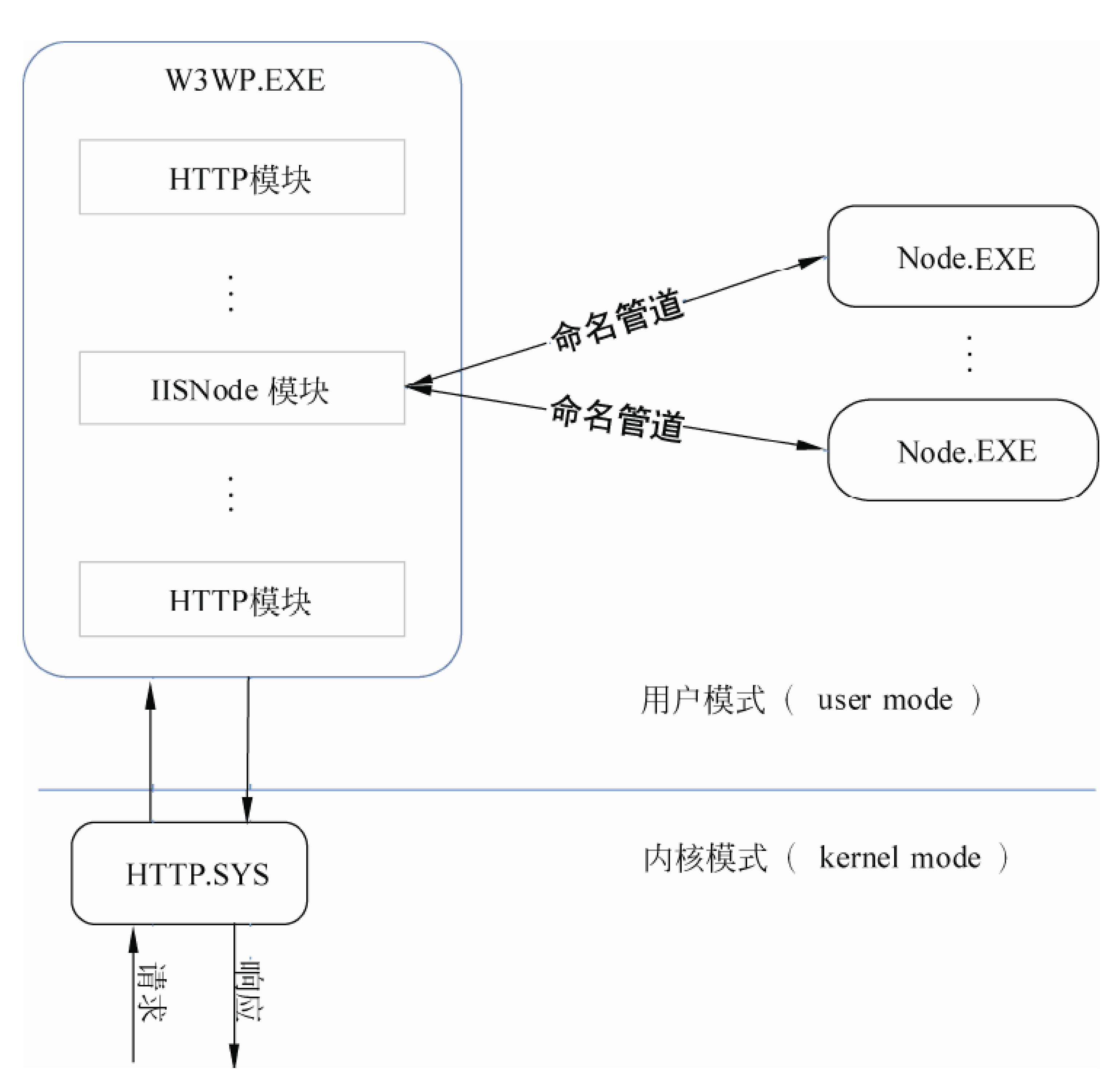


图 4-54　Azure 网站上的 Node.js 架构

在 Azure 网站中，Node.js 请求的处理流程如下：

（1）客户端 HTTP 请求到达 HTTP.SYS（负责处理 HTTP 请求的内核模块）。

（2）HTTP 请求被转发到网站的工作进程 W3WP.EXE。

（3）IIS 将 Node.js 的请求交给 IISNode 模块处理。

（4）根据具体情况，IISNode 模块启动一个新的 Node.EXE，然后将请求转发到新的 Node.EXE；或者将请求转发到一个现有的 Node.EXE。IISNode 与 Node.EXE 之间的通信采用命名管道。

（5）Node.EXE 执行 js 页面，然后将结果返回给 IISNode 模块。

（6）响应通过 HTTP.SYS 发回客户端。

在 Azure 网站上部署 Node.js 时，需要注意下面的事项：

（1）Azure 网站目前只支持 32 位 Node。在管理门户网站将网站设置为 64 位模式，只是将 W3WP.EXE 设置为 64 位，同时将 IISNode 模块设置为 64 位。但是，Node.EXE 仍然是 32 位。

（2）默认 Node.EXE 的实例数目为 1。如果希望运行多个 Node.EXE 的实例，需要修改 IISNode 的配置。

4.5.2 IISNode 配置

IISnode 支持两种配置方法：Web.config 和 iisnode.yml。如果同时使用 Web.config 和 iisnode.yml，那么 iisnode.yml 中的配置会覆盖 Web.config 中的配置。IISNode 的具体定义选项请参考下面的文档：

https://github.com/tjanczuk/iisnode/blob/master/src/config/iisnode_schema.xml

使用 Web.config 配置 IISNode，需要在<system.webServer>节点下添加 iisnode 节点。

```
<system.webServer>
<iisnode ... />
</system.webServer>
```

在这里，简单介绍几个主要的配置选项。

- maxConcurrentRequestsPerProcess

该选项定义了 IISNode 最大并发请求数目。如果请求数超过该选项，IISNode 返回服务不可用错误。该选项的默认值为 1024。

- loggingEnabled

该选项将 stdout 和 stderr 的输出记录到日志文件中。默认值为 false。

- devErrorsEnabled

当发生错误时，该选项控制返回面向客户的“友好的”错误还是面向开发人员的具体的开发错误。打开该选项后，IISNode 返回最近的 64KB 的调试信息，而不是类似服务端错误“请重试。如果问题仍然存在，请联系管理员。”这种面向客户的友好信息。

- nodeProcessCommandLine

该选项指定 Node.EXE 的路径，通常用于选择指定版本的 Node.js。

- nodeProcessCountPerApplication

指定 Node.EXE 实例数目。默认为 1。如果希望运行多个 Node.EXE 实例，需要修改配置。

4.5.3 选择 Node.js 版本

目前 Azure 网站支持如下版本的 Node.js：

- 0.10.18

- 0.10.21
- 0.10.24
- 0.10.26
- 0.10.28
- 0.10.29（目前默认版本）
- 0.10.31
- 0.10.5
- 0.6.17
- 0.6.20
- 0.8.19
- 0.8.2
- 0.8.26
- 0.8.27
- 0.8.28

如果需要使用其他版本的 Node.js，有两种选择：通过指定 Azure 网站应用配置或者通过配置文件。

4.5.3.1　通过 Azure 网站应用配置指定 Node.js 版本

可以通过 WEBSITE_NODE_DEFAULT_VERSION 应用配置指定 Node.js 的版本。下面的步骤通过应用设置指定 Node.js 的版本为 0.10.5。

（1）登录到 Azure 管理门户网站。

（2）单击左侧导航栏中的网站图标。

（3）在右侧网站列表中选择需要加载用户配置文件的网站。

（4）在顶部导航栏，单击“配置”，打开网站的配置页面，定位到“应用设置”区域，如图 4-55 所示，设置应用变量 WEBSITE_NODE_DEFAULT_VERSION=0.10.5。

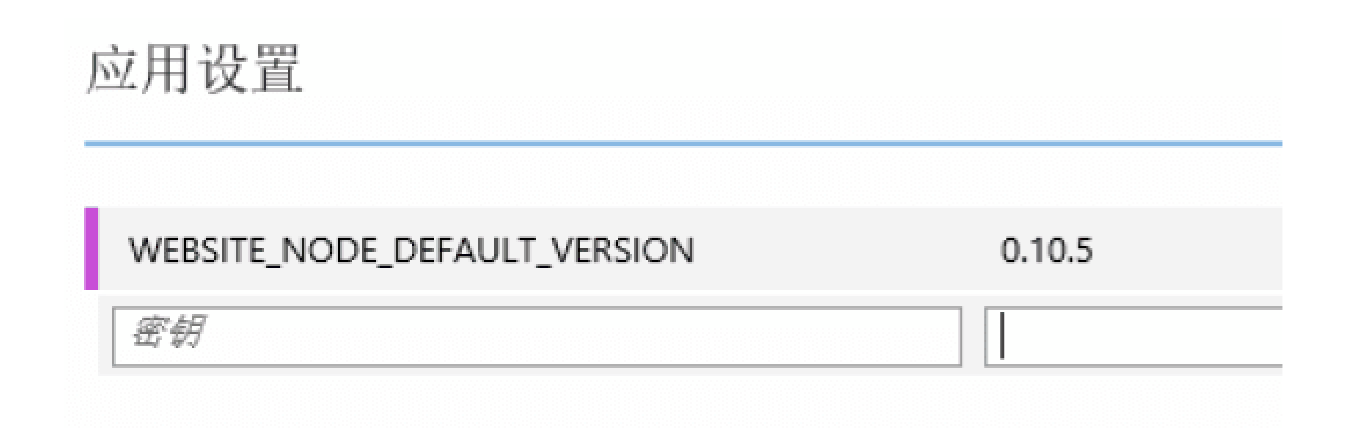


图 4-55　指定 Node.js 版本

（5）单击底部命令栏中的“保存”按钮。

（6）单击底部命令栏中的“重新启动”按钮，重启网站。

4.5.3.2　通过配置文件指定 Node.js 版本

可以通过 nodeProcessCommandLine 选项指定 Node.js 的版本。比如，下面的例子指定 0.10.31 版本。

使用 Web.config：

```
<system.webServer>
    <iisnode
    nodeProcessCommandLine="D:\Program Files (x86)\nodejs\0.10.31\node.exe"
    />
</system.webServer>
```

或者使用 IISNode.yml：

```
nodeProcessCommandLine: "D:\Program Files (x86)\nodejs\0.10.31\node.exe"
```

4.5.4 节将通过一个具体的例子演示如何选择 Node.js 的版本。

4.5.4 利用 Visual Studio 开发和部署 Node.js 应用

关于 Node.js 的开发，在 Azure 平台上与本地 Windows 平台并无太大区别，本节不做过多的介绍。本节主要介绍微软公司的开源项目 Node.js Tools for Visual Studio（NTVS）。

NTVS 支持 Visual Studio 2012 和 Visual Studio 2013。使用 NTVS，可以借助 Visual Studio 的编辑器和调试器开发、调试 Node.js 应用。NTVS 提供了编辑、智能提示、分析、调试（本地和远程）功能，并可以将 Node.js 发布到 Azure 网站和云服务。

可以在 codeplex 网站下载该工具以及源代码：

https://nodejstools.codeplex.com/

下面通过一个简单的例子来演示如何利用 NTVS 工具开发部署 Node.js 应用。

（1）通过上面的网址下载安装该工具。

（2）运行 Visual Studio，选择“新建”→“项目”，如图 4-56 所示，选择 Node.js 项目模板，然后选择空白 Azure Node.js Web 应用。将应用命名为 DisplayVersion，单击“确定”创建应用。

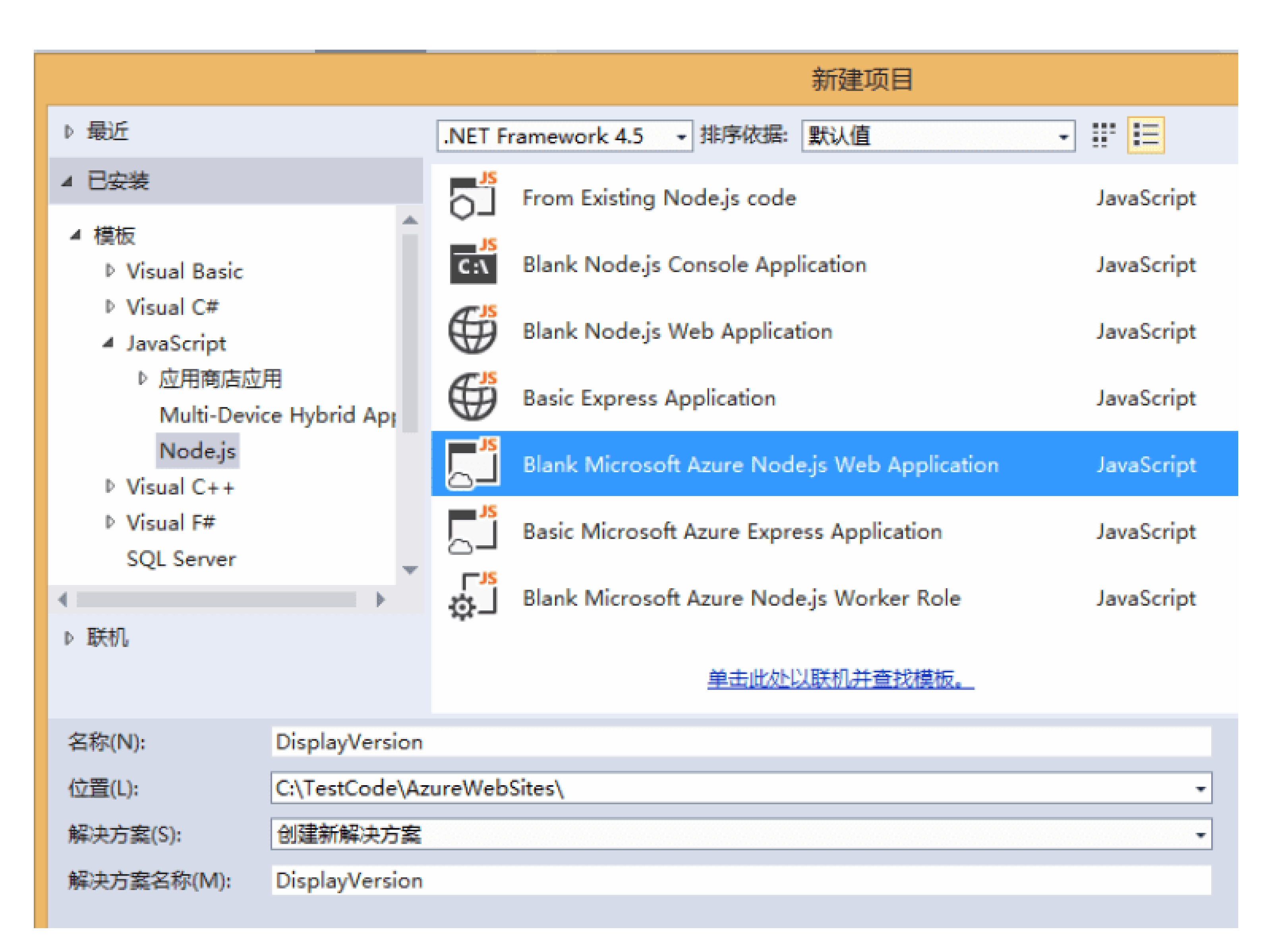


图 4-56 新建 Node.js 项目

（3）在“解决方案资源管理器”中，找到并打开 server.js 文件，然后找到下面的代码：

```
res.end('Hello World');
```

将其修改为

```
res.end('Node Version: ' + process.version);
```

（4）在“解决方案资源管理器”中，右击项目，选择“发布”，如图 4-57 所示，将项目发布到 Azure 网站中。

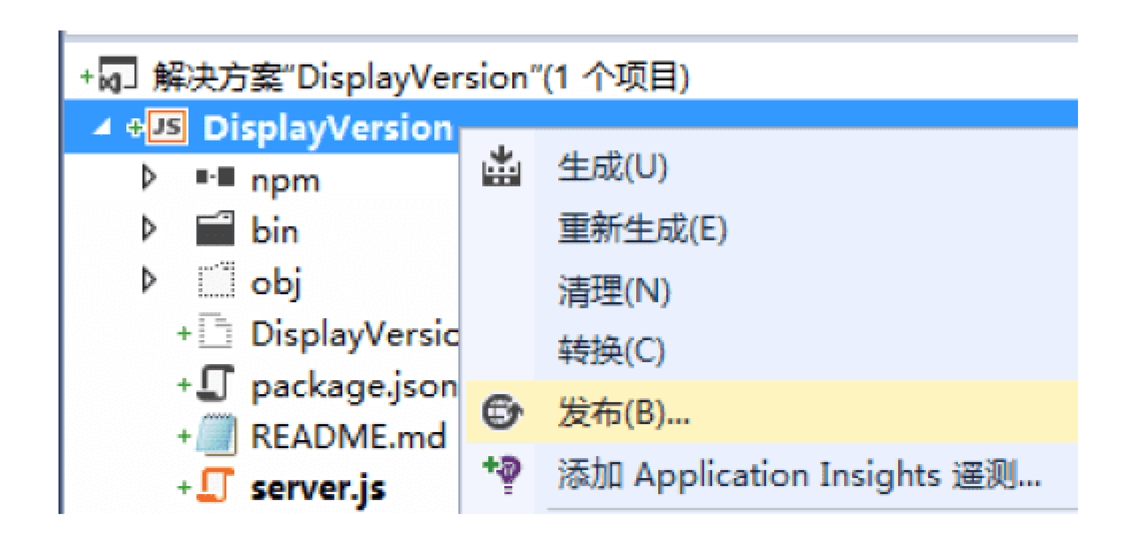


图 4-57　发布 Node.js 项目

（5）在“发布 Web”对话框中，3 个选项都可以将 Node.js 应用发布到 Azure 网站。方便起见，在这里选择 Microsoft Azure 网站，如图 4-58 所示。用 Azure 订阅登录后，可以选择发布到现有网站或者新建一个网站。

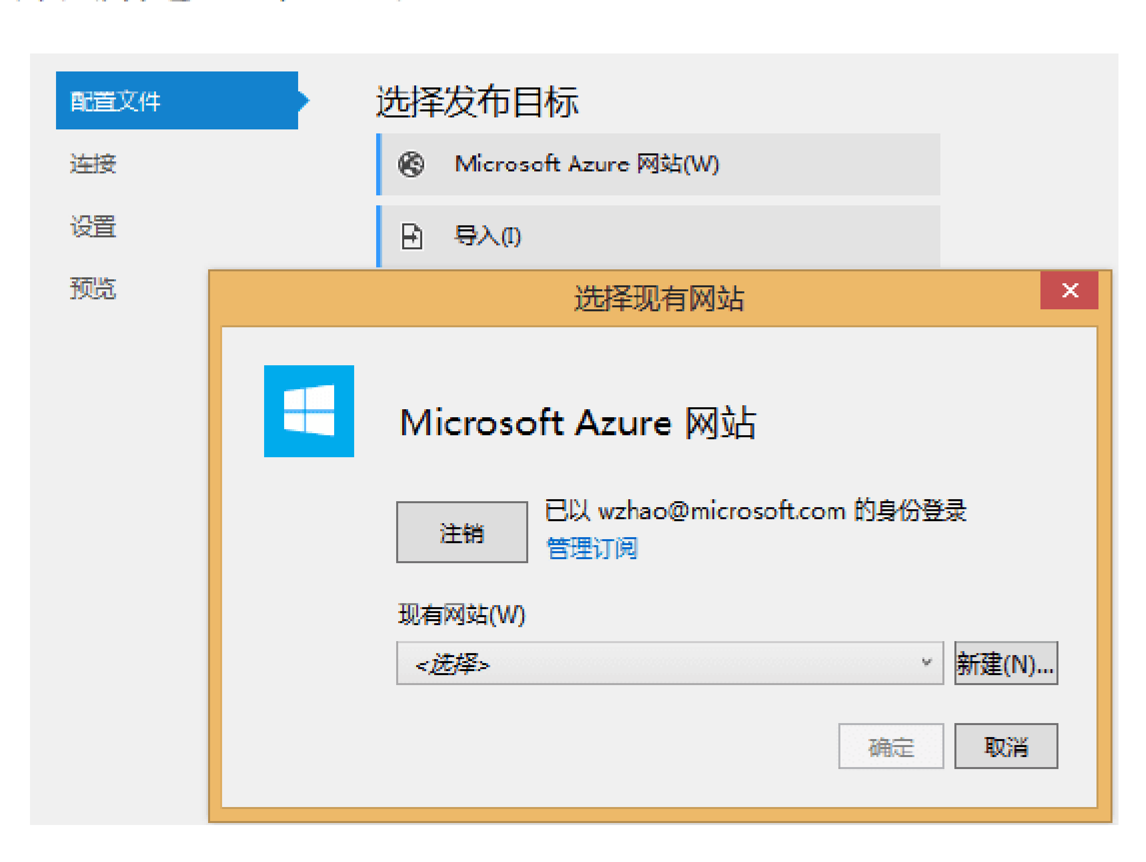


图 4-58　选择网站

（6）在“发布 Web”对话框中，单击“发布”。将 Node.js 应用发布到 Azure 网站。如图 4-59 所示，Node.js 的版本为 0.10.29。



图 4-59　Node.js 默认版本

（7）回到 Visual Studio 中，打开 Web.config 文件。找到下面的代码：

```
<iisnode watchedFiles="web.config;*.js" />
```

将其修改为

```
<iisnode watchedFiles="web.config;*.js"
    nodeProcessCommandLine="D:\Program Files (x86)\nodejs\0.10.31\node.exe"
/>
```

（8）重新部署，此时再次访问网页，会发现 Node 的版本变为 0.10.31。

4.5.5 利用 Visual Studio 调试 Node.js 应用

Node.js Tools for Visual Studio（NTVS）支持在 Visual Studio 中实时调试部署到 Azure 网站的 Node.js 应用。本节以 4.5.4 节的项目为例，介绍如何在 Visual Studio 中在线调试 Node.js 应用。

（1）部署 Debug 版本的 Node.js 到 Azure 网站。为此，在“发布 Web”对话框的配置页面选择 Debug 版本，如图 4-60 所示。



图 4-60 部署 Debug 版本的 Node.js

在 Debug 版本的 Web.config 中，定义了如下 IISNode 配置项：

```
    <iisnode loggingEnabled="true"
            devErrorsEnabled="true"
            interceptor="--debug"
            xdt:Transform="SetAttributes"
/>
```

同时，定义了一个调试的 handler 和 URL rewrite 的规则。具体的定义可以参考项目中的 Web.Debug.config 文件。

（2）登录到管理门户网站，在网站的配置页面，如图 4-61 所示，打开 WebSocket 选项。



图 4-61 打开网站 WEB SOCKET 选项

（3）与调试本地程序相同，在 Visual Studio 中设置代码断点。

（4）在服务资源管理器中，连接到 Azure 订阅。如图 4-62 所示，右击要调试的网站，选择 Attach Debugger(Node.js)，启动调试。

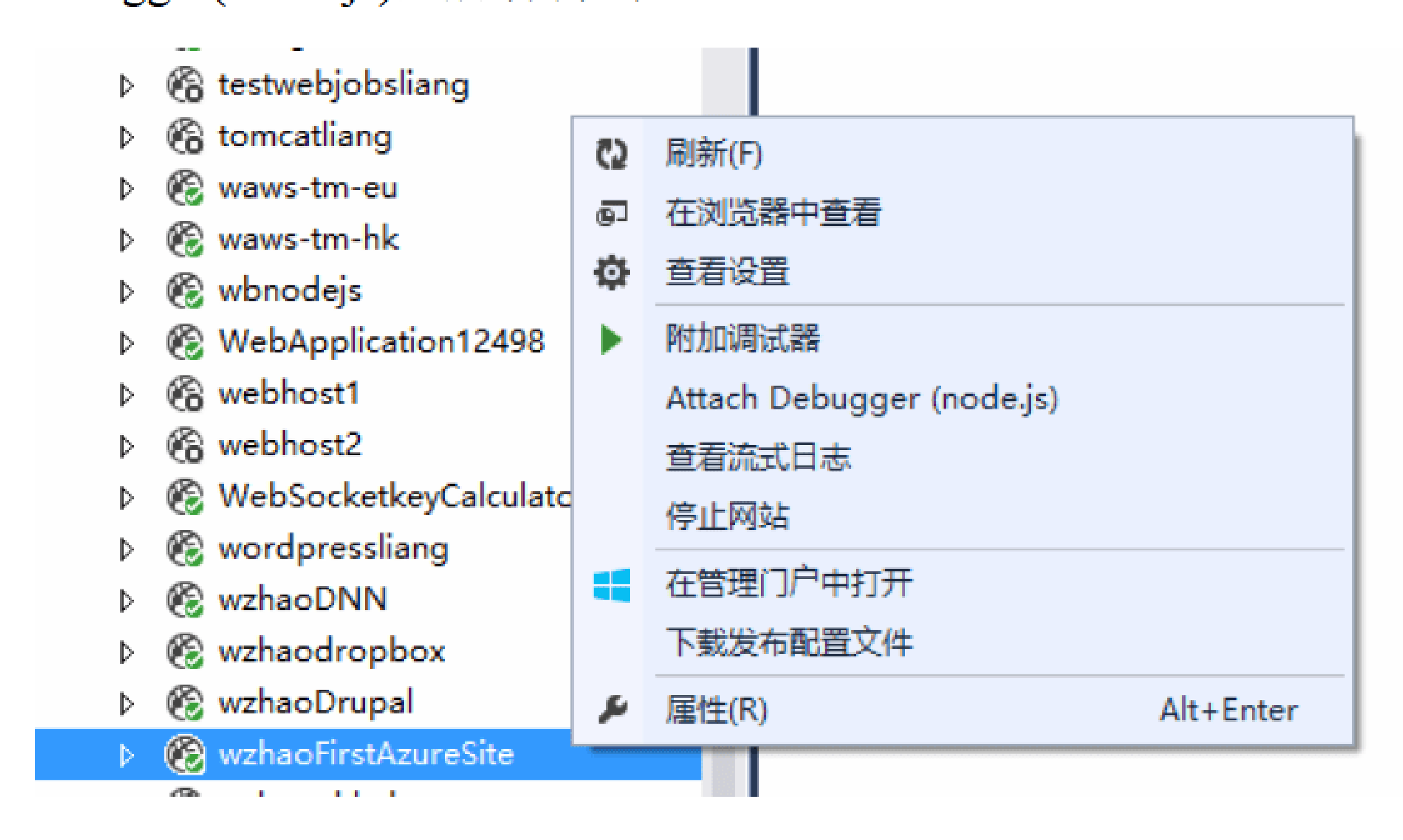


图 4-62　附加 Node.js 调试器

（5）调试启动后，当设置断点的代码被执行时，Visual Studio 调试器自动介入，如图 4-63 所示。

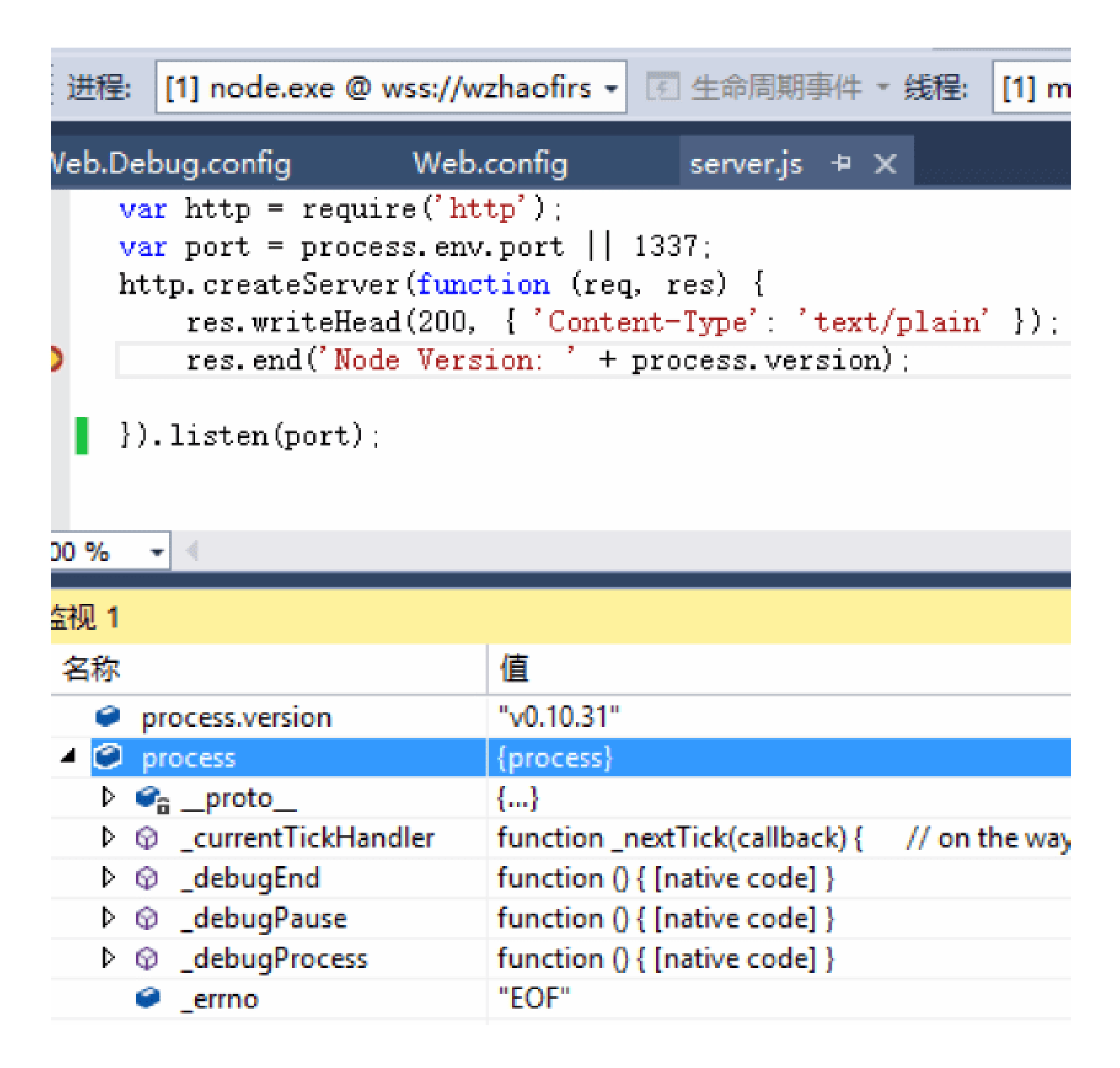


图 4-63　在 Visual Studio 中调试 Node.js

4.6　Azure 网站应用设置

Azure 网站允许开发人员通过键-值（key-value）字符串对的形式指定网站应用配置。这些配置与对应的网站绑定在一起。在运行时，Web 应用可以通过环境变量或者 ASP.NE

应用配置项的方式访问这些应用设置。

应用配置作为网站配置的一部分保存在 Azure 网站的运行时数据库中，而不是保存在网站的文件中，比如 Web.config。在存储敏感信息，比如数据库连接字符串时，这样比保存在 Web.config 中更加安全。

4.6.1 使用应用设置

本章前几节中介绍了如何通过应用配置指定 Node.js 的版本，以及通过应用配置加载用户配置文件。下面介绍如何在自己的代码中利用应用配置。

4.6.1.1 配置应用设置

（1）登录到 Azure 管理门户网站。

（2）单击左侧导航栏中的网站图标。

（3）在右侧网站列表中选择需要加载用户配置文件的网站。

（4）在顶部导航栏，单击“配置”，打开网站的配置页面，定位到应用设置区域，如图 4-64 所示，设置应用需要的配置。下面的例子配置了两个应用设置：MAX_CACHE_SIZE 和 CACHE_TTL。

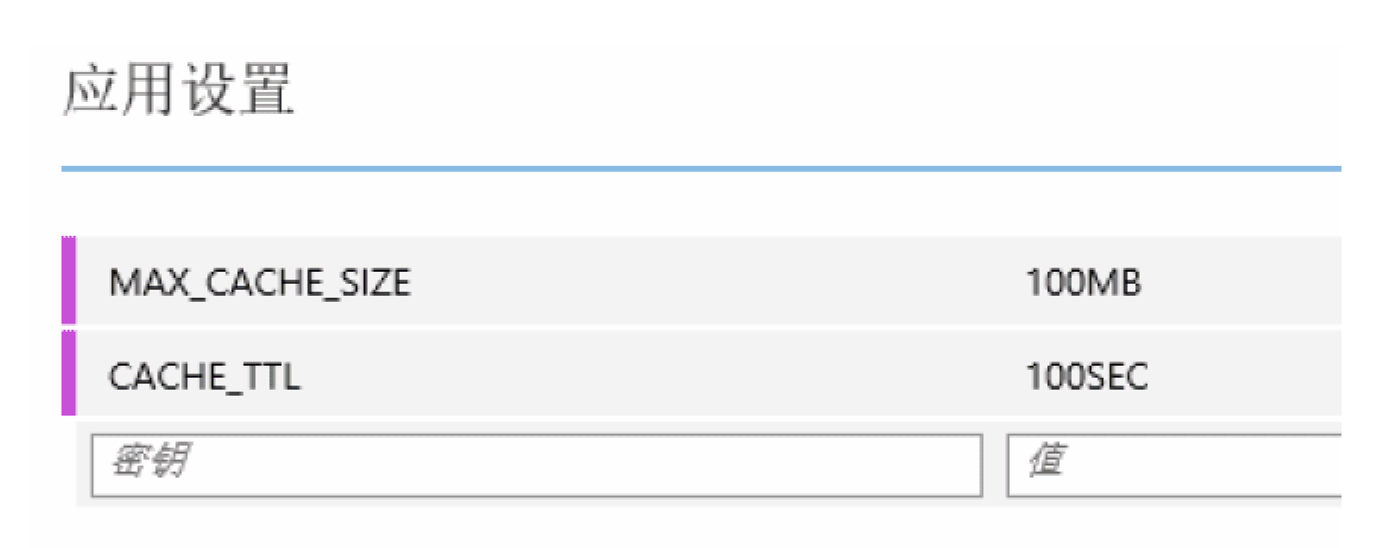


图 4-64 配置应用设置

注意：Azure 网站以字符串类型保存应用设置。

（5）单击底部命令栏中的“保存”按钮。

（6）单击底部命令栏中的“重新启动”按钮，重启网站。

4.6.1.2 通过环境变量读取应用配置

当网站启动时，Azure 网站自动将应用配置插入到环境变量中。在运行时，开发人员可以通过环境变量的方式读取指定的网站应用设置。

下面的 PHP 示例通过 getenv API 读取指定的应用配置：

```
<?php
  $max_cache_size = getenv('MAX_CACHE_SIZE');
  echo $max cache size;

  echo '<br>' ;
```

```
  $cache_ttl = getenv('CACHE_TTL');
  echo $cache_ttl;
?>
```

下面的 ASP.NET Web Page（.cshtml）演示了如何在 ASP.NET 中通过环境变量读取应用配置：

```
<html>
<body>
   <div>
      <p>MAX_CACHE_SIZE:
          @Environment.GetEnvironmentVariable("MAX_CACHE_SIZE")
</p>
       <p>CACHE_TTL:
@Environment.GetEnvironmentVariable("CACHE_TTL")
</p>
    </div>
</body>
</html>
```

环境变量的名称可以是应用配置名称，比如 MAX_CACHE_SIZE，也可以是 APPSETTING_<配置名称>，比如 APPSETTING_MAX_CACHE_SIZE 作为环境变量的名称读取，两者皆可。

4.6.1.3　通过 ASP.NET 读取应用设置

除了将网站定义的应用设置加入到环境变量中，Azure 网站同时将网站的应用设置添加到 ASP.NET 的 AppSettings 配置列表中。因此，在 ASP.NET 中，除了可以通过环境变量读取应用设置外，还可以通过 ASP.NET 配置管理器读取网站的应用设置。该方法与读取 web.config 中的 appsettings 元素中定义的应用设置完全相同。下面的 ASP.NET Web Page（.cshtml）演示了如何在 ASP.NET 中通过配置管理器读取网站定义的应用配置。

```
<html lang="en">
  <body>
     <div>
        <p>MAX_CACHE_SIZE:
            @System.Configuration.ConfigurationManager.AppSettings["MAX_
            CACHE_SIZE"]
        </p>
        <p>CACHE_TTL:
            @System.Configuration.ConfigurationManager.AppSettings
            ["CACHE _TTL"]
        </p>
     </div>
  </body>
</html>
```

4.6.2　数据库连接字符串

除了应用设置外，Azure 网站还允许开发人员为网站指定数据库连接字符串。通常，ASP.NET 开发人员将数据库连接字符串保存在 web.config 文件中，PHP 开发人员将数据库连接字符串保存在.ini 文件中。相比保存在文件中的方式（web.config/.ini），将数据库连接字符串与网站的其他配置一起保存在网站的后台数据库中是更为安全的方式。

Azure 网站中，数据库连接字符串与应用设置完全相同，都是通过键-值的方式指定。读取数据库连接字符串的方式也完全相同。唯一不同的是，Azure 网站和应用可以区分连接字符串，进行特殊处理。比如，默认情况下，连接字符串中的用户密码等信息会在显示的时候以星号替代。

Azure 网站支持 4 种方式的数据库连接字符串，并分别给予不同的前缀进行区别：

- SQL Server，前缀为 SQLCONNSTR_，用于指定运行在企业内部数据中心或者 Azure 虚拟机上的，客户自己安装、运行、维护的 SQL 服务器。
- SQL 数据库，前缀为 SQLAZURECONNSTR_，用于指定 SQL Azure 连接字符串。
- MySQL 数据库，前缀为 MYSQLCONNSTR，用于指定 MySQL 数据库连接字符串。
- 自定义链接字符串，前缀为 CUSTOMCONNSTR，用户指定客户自定义链接字符串，比如，可以指定 Azure 存储连接字符串。

在本节中，详细介绍如何配置和读取数据库连接字符串。

4.6.2.1　配置数据库连接字符串

（1）登录到 Azure 管理门户网站。

（2）单击左侧导航栏中的网站图标。

（3）在右侧网站列表中选择需要加载用户配置文件的网站。

（4）在顶部导航栏，单击“配置”，打开网站的配置页面，定位到连接字符串区域，设置应用需要的数据库连接字符串。图 4-65 指定了一个 SQL Azure 的连接字符串。

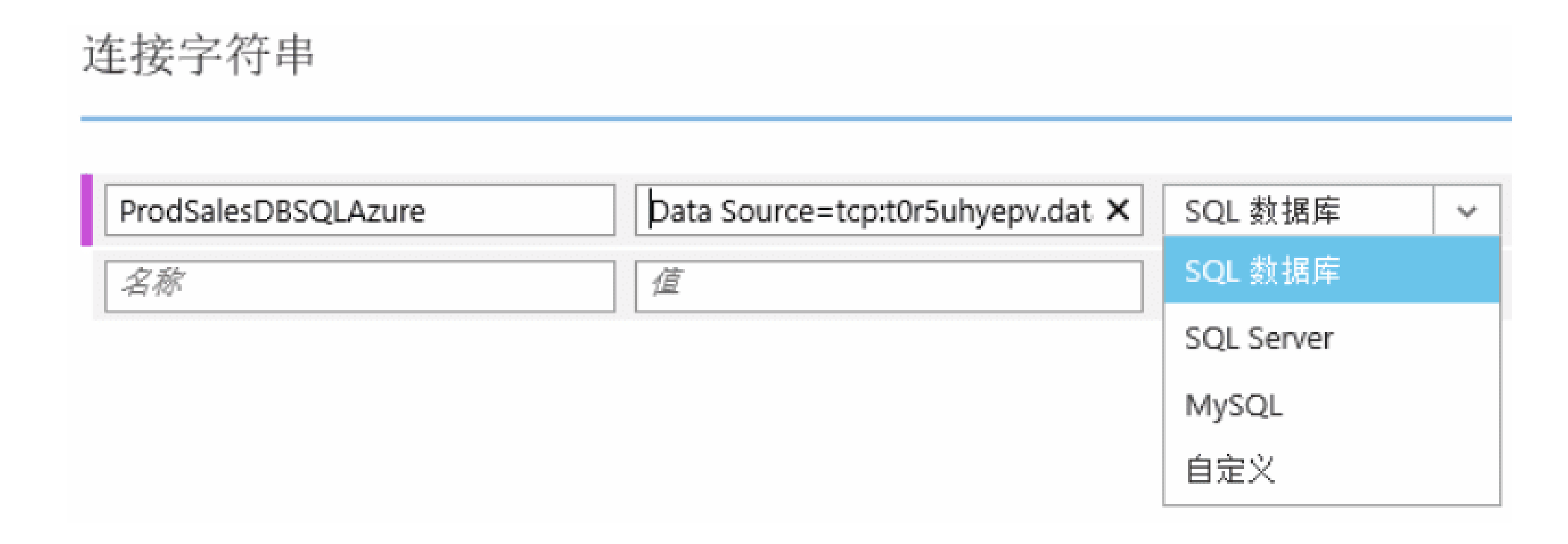


图 4-65　指定数据库连接字符串

（5）单击底部命令栏中的“保存”按钮。

（6）单击底部命令栏中的“重新启动”按钮，重启网站。

（7）此时，数据库连接字符串自动被隐藏。要查看连接字符串，请单击“显示连接字

符串”链接，如图 4-66 所示。

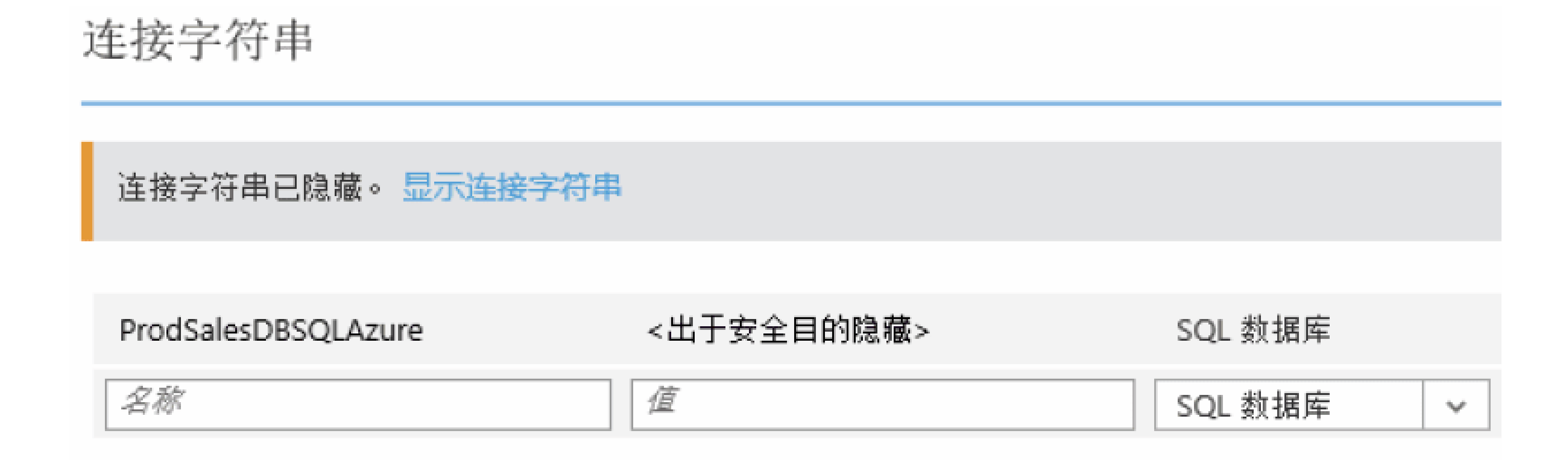


图 4-66　隐藏的数据库连接字符串

4.6.2.2　通过环境变量读取数据库连接字符串

与网站应用设置相同，当网站启动时，Azure 网站自动将数据库连接字符串插入到环境变量中。环境变量的名称为“数据库类型前缀_连接字符串名称”。比如，图 4-66 中的数据库连接字符串对应的环境变量名称为 SQLAZURECONNSTR_ProdSalesDBSQLAzure。在运行时，开发人员可以通过环境变量读取指定的网站应用设置。

下面的 PHP 示例通过 getenv API 读取指定的数据库连接字符串。

```
<?php
  $conn_str_env = getenv('SQLAZURECONNSTR_ProdSalesDBSQLAzure');
  echo $conn_str_env;
?>
```

下面的 ASP.NET Web Page（.cshtml）演示了如何在 ASP.NET 中通过环境变量读取数据库连接字符串：

```
<html>
  <body>
     <div>
        <p>CONN STR:
              @Environment.GetEnvironmentVariable(
                  "SQLAZURECONNSTR_ProdSalesDBSQLAzure")
          </p>
      </div>
  </body>
</html>
```

4.6.2.3　使用 ASP.NET 配置管理器读取连接字符串

除了将网站定义的数据库连接字符串加入到环境变量中之外，Azure 网站同时将网站的数据库连接字符串添加到 ASP.NET 的 ConnectionStrings 配置列表中。因此，在 ASP.NET 中，除了可以通过环境变量读取数据库连接字符串外，还可以通过 ASP.NET 配置管理器读取连接字符串。该方法与读取 web.config 中的 ConnectionStrings 元素中定义的数据库连接字符串完全相同。下面的 ASP.NET Web Page（.cshtml）演示了如何在 ASP.NET 中通过配

置管理器读取网站配置的数据库连接字符串：

```
<html lang="en">
  <body>
    <div>
      <p>CONN STR:
            @System.Configuration.ConfigurationManager.ConnectionStrings
            ["ProdSalesDBSQLAzure"]
       </p>

     </div>
  </body>
</html>
```

4.6.3 运行时自动更新

在 4.6.1.3 小节中提到，网站启动时，Azure 网站自动将开发人员定义的应用设置加入到环境变量中，除此之外，Azure 网站同时将网站的应用设置添加到 ASP.NET 的 appSettings 配置列表中。

如下所示，如果在 web.config 的 appSettings 元素中已经定义了名为 MAX_CACHE_TTL 的应用设置，那么在网站启动时，Azure 网站上定义的 MAX_CACHE_TTL 设置自动覆盖 web.config 中定义的设置。

```
<configuration>
    <appSettings>
       <add key="MAX CACHE TTL" value="10MB" />
    </appSettings>
</configuration>
```

与应用设置相同，除了将网站定义的数据库连接字符串加入到环境变量中之外，Azure 网站同时将网站的数据库连接字符串添加到 ASP.NET 的 connectionStrings 配置列表中。

如下所示，如果在 web.config 中定义了命名为 ProdSalesDBSQLAzure 的数据库连接字符串，Azure 网站自动使用在管理门户网站上配置的相同名称的数据库连接字符串覆盖在 web.config 中定义的连接字符串。

```
<configuration>
  <connectionStrings>
     <add name="ProdSalesDBSQLAzure"
        providerName="System.Data.SqlClient"
        connectionString="server=myserver;database=SalesDB;
             uid=<user name>;pwd=<secure password>" />
  </connectionStrings>
</configuration>
```

运行时自动更新功能可以解决某些特定的问题。比如，如果开发人员在部署网站时不小心将测试环境的 web.config 配置部署到生产环境，Azure 网站自动使用在网站中定义的设置将 web.config 中的配置覆盖，从而避免生产环境宕机。

4.7　使用 Visual Studio Online（Monaco）在线编辑代码

Visual Studio Online（Monaco）是一个轻量级的在线代码编辑环境。Monaco 作为 Azure 网站的一个扩展，与 Azure 网站无缝集成在一起。可以在浏览器中通过 Monaco 在线编辑网站代码，包括 PHP、Node.js、ASP.NET 等。同时，Monaco 支持 Git 和 Visual Studio Online 版本管理。在本节，简单介绍如何利用 Monaco 在线编辑代码。

4.7.1　打开在线编辑功能

Azure 网站中的在线编辑功能默认是关闭的。要使用在线编辑功能，需要通过下面的步骤打开该功能。

（1）登录到 Azure 管理门户网站。

（2）单击左侧导航栏中的网站图标。

（3）在右侧网站列表中选择需要打开在线编辑功能的网站。

（4）在顶部导航栏，单击“配置”，打开网站的配置页面，如图 4-67 所示，打开“在 VISUAL STUDIO ONLINE 中在线编辑”功能。



图 4-67　打开在线编辑功能

（5）单击底部命令栏的“保存”按钮。

（6）在顶部导航栏中单击“仪表板”，在“速览”区域，如图 4-68 所示，显示“在 Visual Studio Online 中编辑”链接，该链接指向 https://<sitename>.scm.azurewebsites.net/dev。



图 4-68　在线编辑链接

（7）单击该链接，即可在线编辑网站代码。Monaco 作为 Azure 网站的一个扩展，需要提供 Azure 订阅的用户名和密码登录。

4.7.2　通过 Monaco 在线编辑代码

Monaco 是一个基于 HTML5 的网站，如图 4-69 所示，界面非常简洁、直观，主要包括以下几个部分：

（1）顶部信息栏：包含 Monaco 的标识、网站名称、用户名称、Monaco 设置选项、帮助和反馈等。

（2）右侧导航栏：以图标的形式列出 Monaco 提供的基本功能，接下来会详细介绍这些功能。

（3）文件导航树：与文件浏览器一样，该区域以树状形式显示网站文件。

（4）文件编辑区域：该区域用于编辑代码。

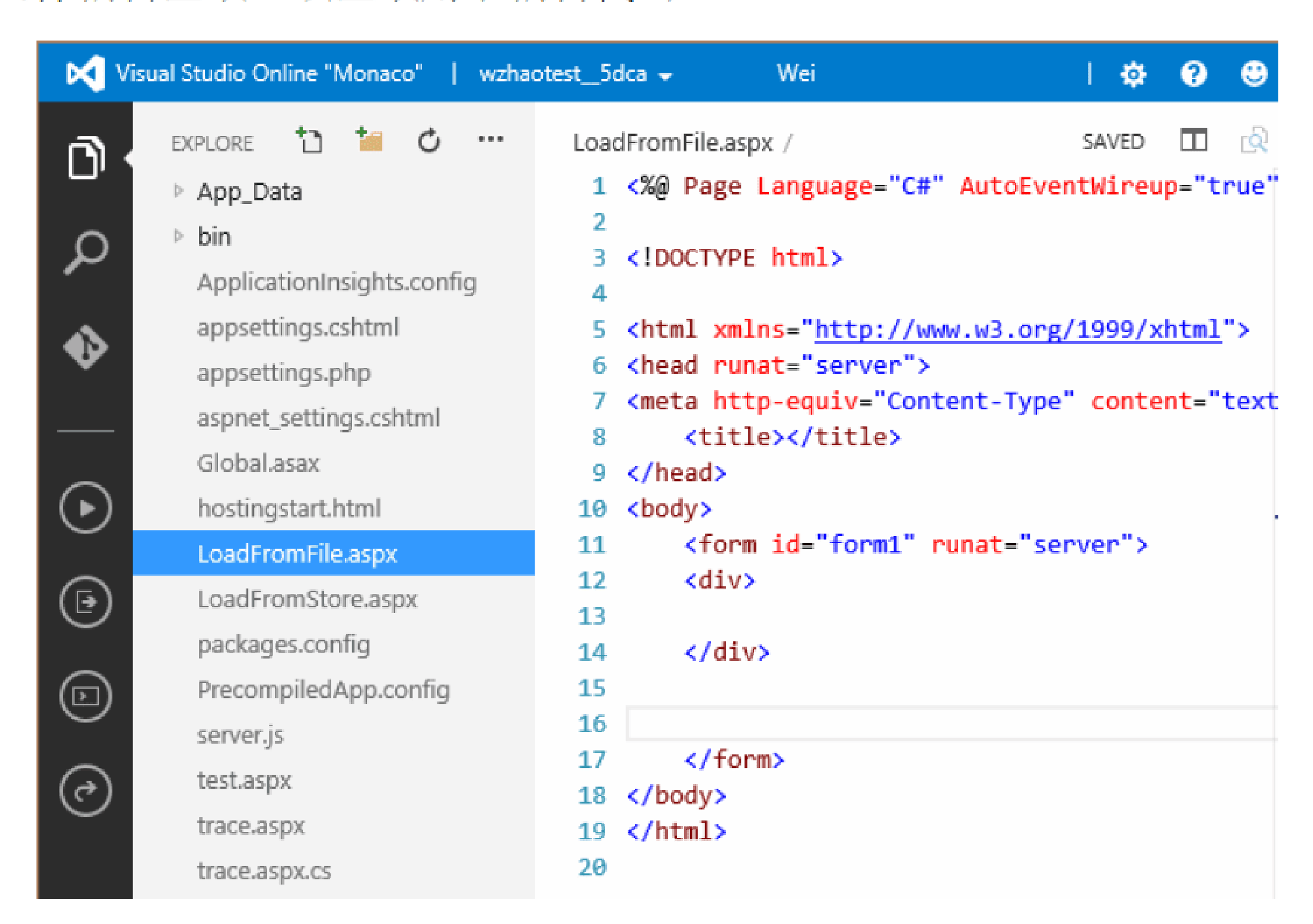


图 4-69　代码编辑区域

4.7.2.1　网站文件操作

Monaco 以树状的形式显示网站文件。通过 Monaco，可以方便地操作文件。

1. 新建文件

在 Monaco 中，要创建新的文件，只需单击文件树顶部的新建文件图标。

2. 新建文件夹

要创建新的文件夹，在 Monaco 文件浏览器单击顶部的新建文件夹图标。

3．上传文件

Monaco 支持两种方式上传文件。如图 4-70 所示，可以单击文件树顶部的“…”图标，然后选择 Upload files。同时 Monaco 支持文件拖曳操作，可以通过从本地文件浏览器中直接拖曳文件到 Monaco 的方式上传文件。

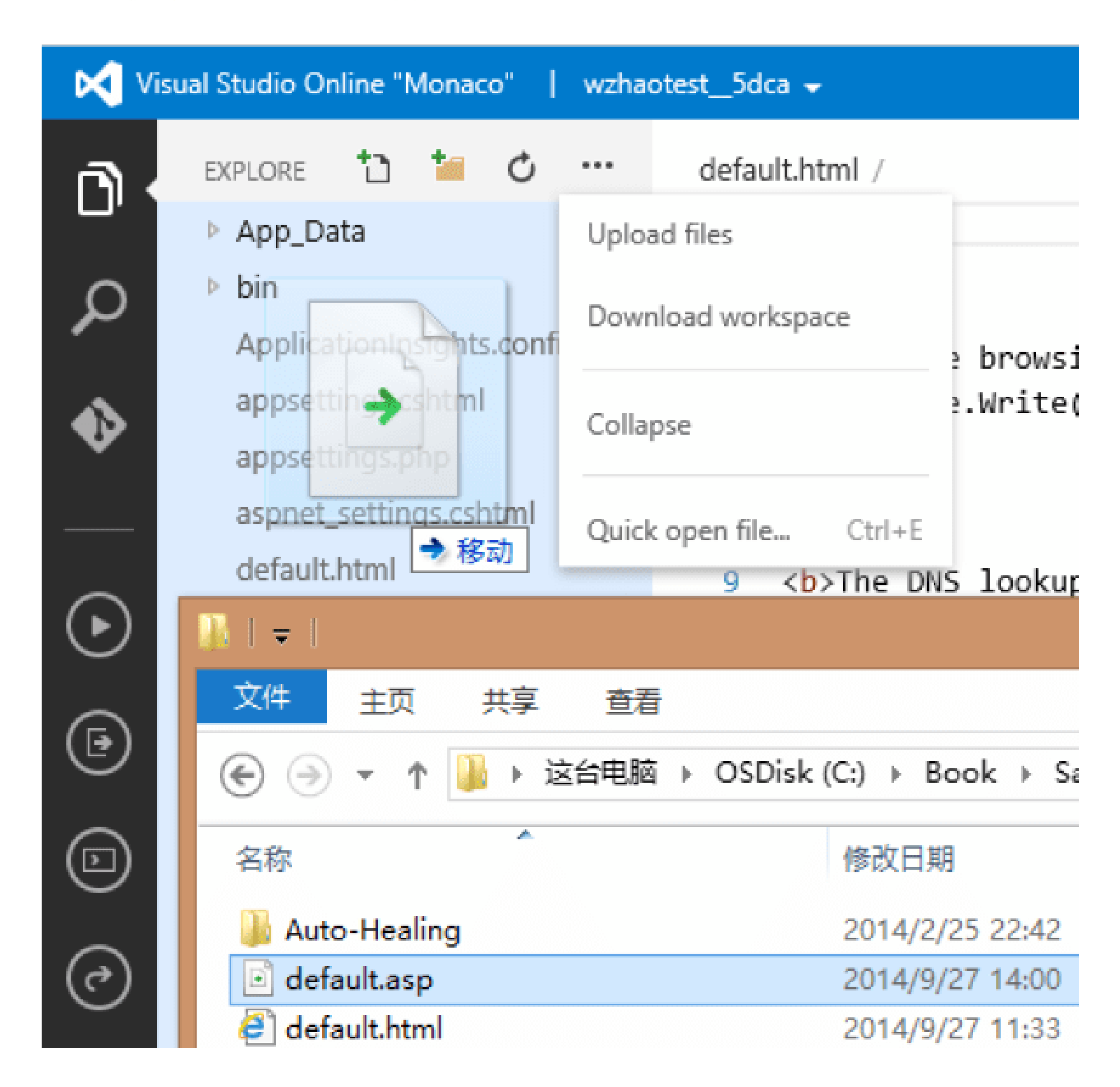


图 4-70　在 Monaco 中通过拖曳上传文件

4．下载文件

如图 4-70 所示，单击文件树顶部的“···”图标，然后选择 Download workspace。

5．删除文件

在 Monaco 中，要删除文件或者文件夹，只需选中要删除的文件或者文件夹，然后按删除（Delete）键。或者，右击要删除的文件，然后选择 Del 命令。

6．复制文件

Monaco 中复制文件与本地文件浏览器完全相同。选中要复制的文件，按 Ctrl+C 快捷键，再选择目标文件夹，然后按 Ctrl+V 快捷键。或者通过右击要复制的文件，选择 Copy 命令，再右击目标文件夹，选择 Paste 命令。

7．打开文件进行编辑

在文件导航树中单击要编辑的文件，Monaco 自动在编辑区域显示文件内容。此时，即可在文件编辑区编辑代码。

如果网站有很多文件，通过文件导航树查找文件效率比较低。如图 4-71 所示，可以按

Ctrl+E 快捷键，或者单击编辑区顶部的文件名称，此时 Monaco 显示文件搜索框，输入要编辑的文件名即可快速打开该文件进行编辑。

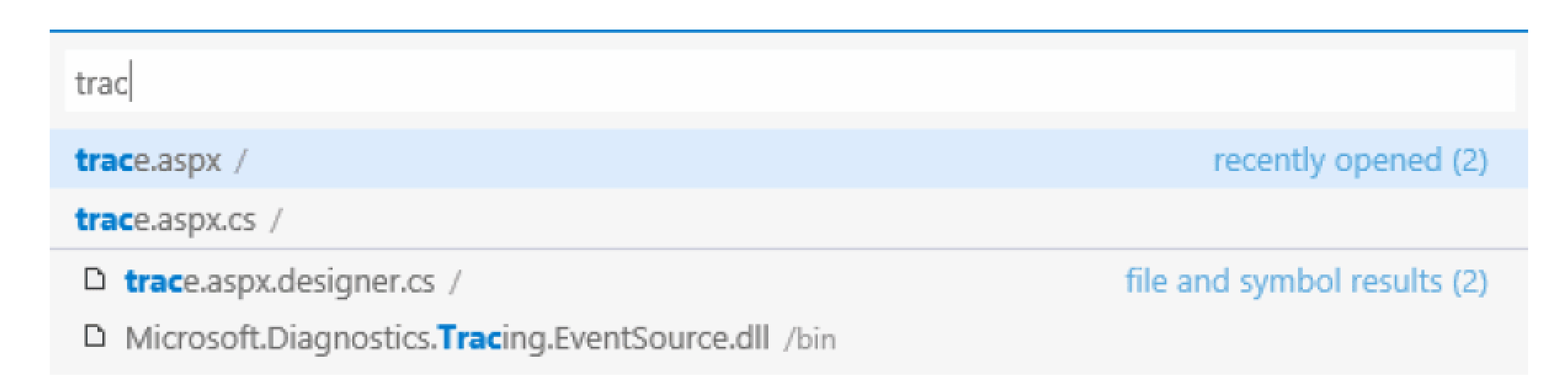


图 4-71 Monaco 中快速打开文件

4.7.3 集成源代码管理

Monaco 支持源代码版本管理，本节将演示如何连接到 Visual Studio Online 实现代码版本管理。

（1）如图 4-72 所示，在 Monaco 中，单击左侧功能栏的 GIT 图标，然后选择 Connect to Visual Studio Online。

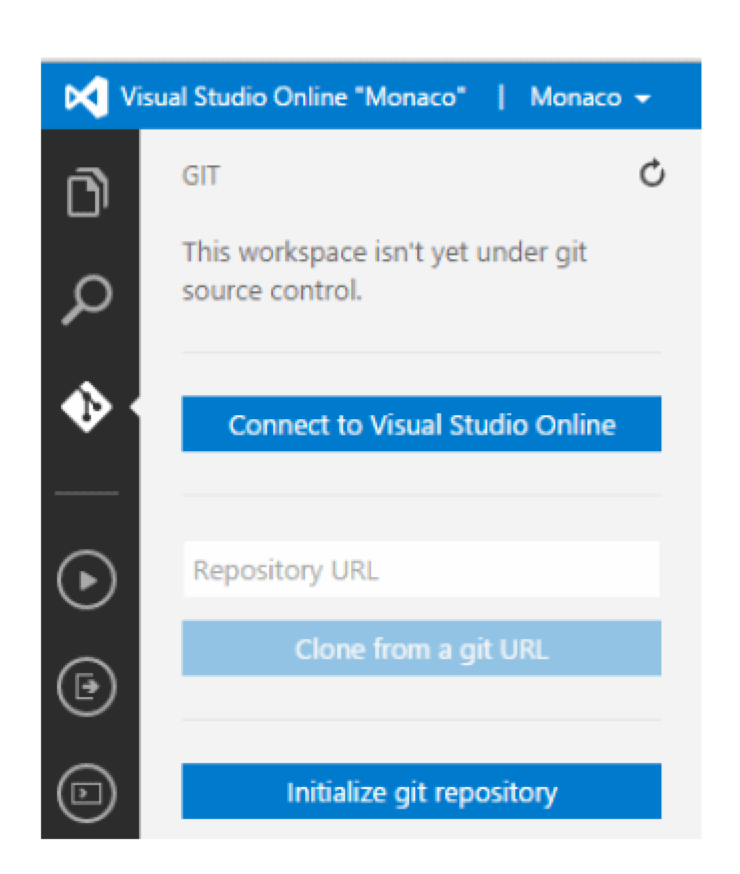


图 4-72 在 Monaco 中集成代码管理

（2）如图 4-73 所示，输入 Visual Studio Online 账户名称，然后单击 Connect。在连接过程中，要求输入账户密码进行验证。

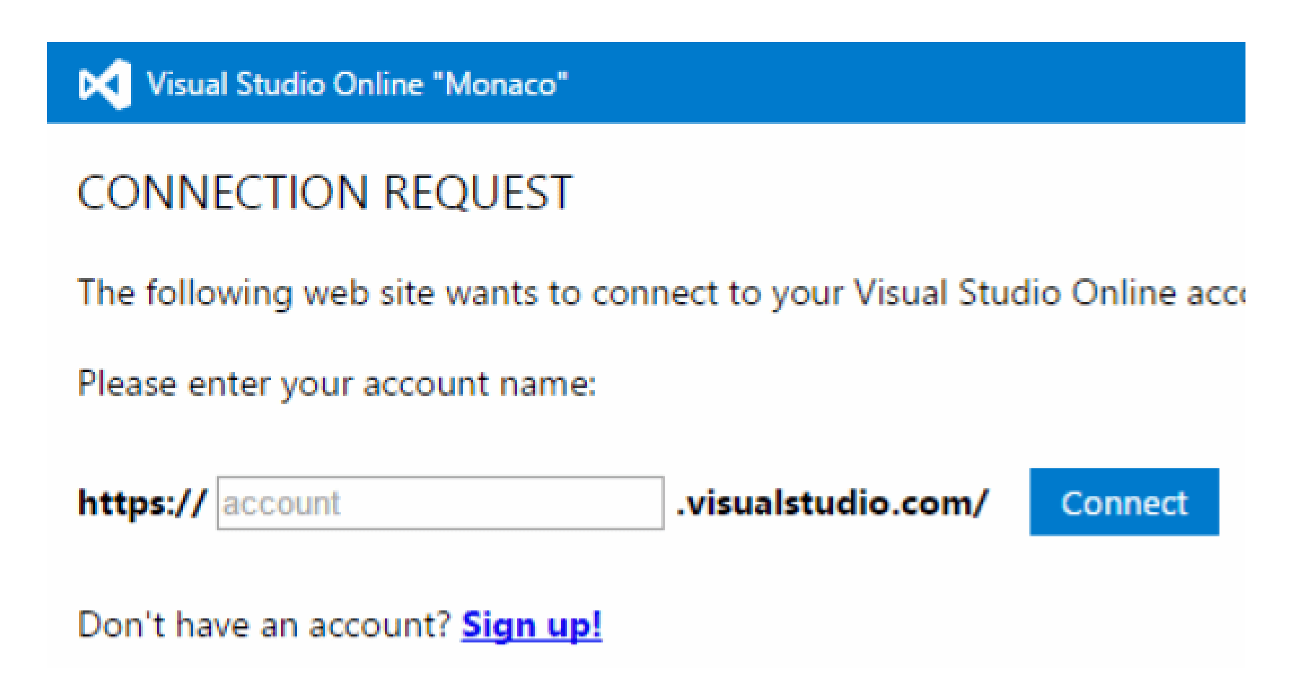


图 4-73 连接到 Visual Studio Online

（3）连接后，如图 4-74 所示，下拉列表中会显示 Visual Studio Online 中所有使用 GIT 进行版本控制的项目。选择要连接的项目，然后选择 Clone from VS Online repository。

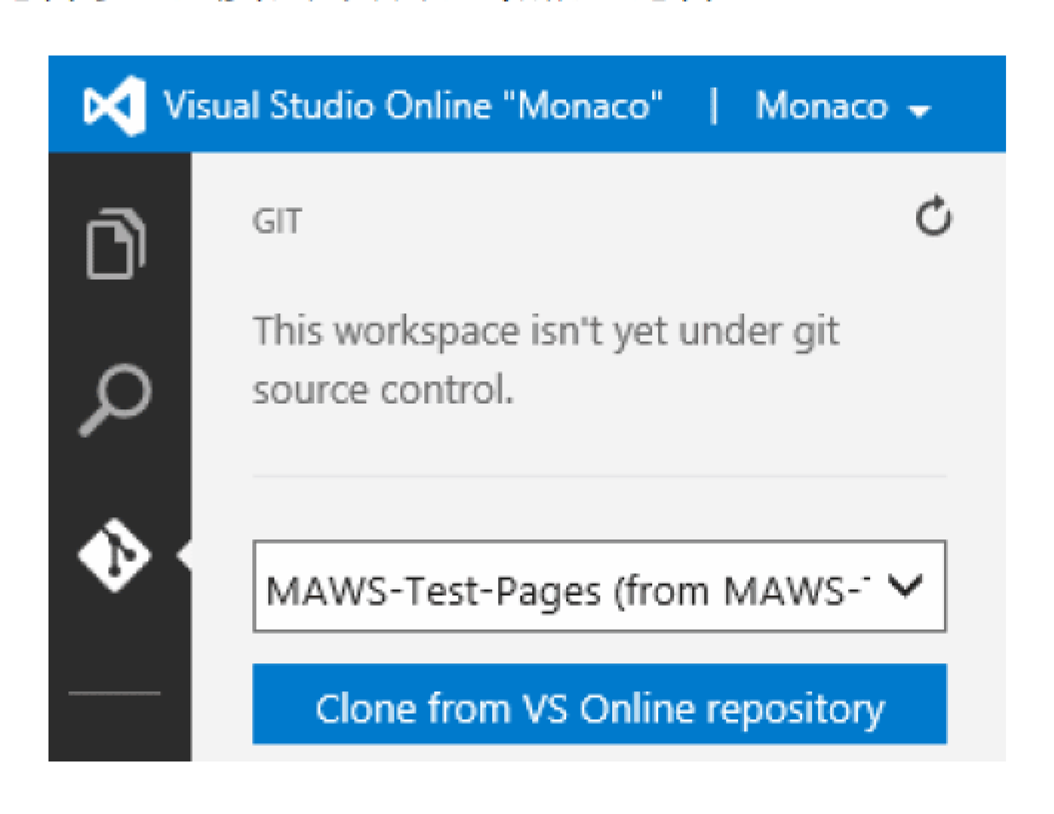


图 4-74　选择要连接的项目

（4）Monaco 运行 Git Clone 命令从 Visual Studio Online 代码库中克隆项目。

（5）如图 4-75 所示，修改、删除、添加代码后，Monaco 会标识更新过的文件，在 Commit message 输入框中输入代码修改描述信息后，单击 Commit All 可以提交修改后的代码。

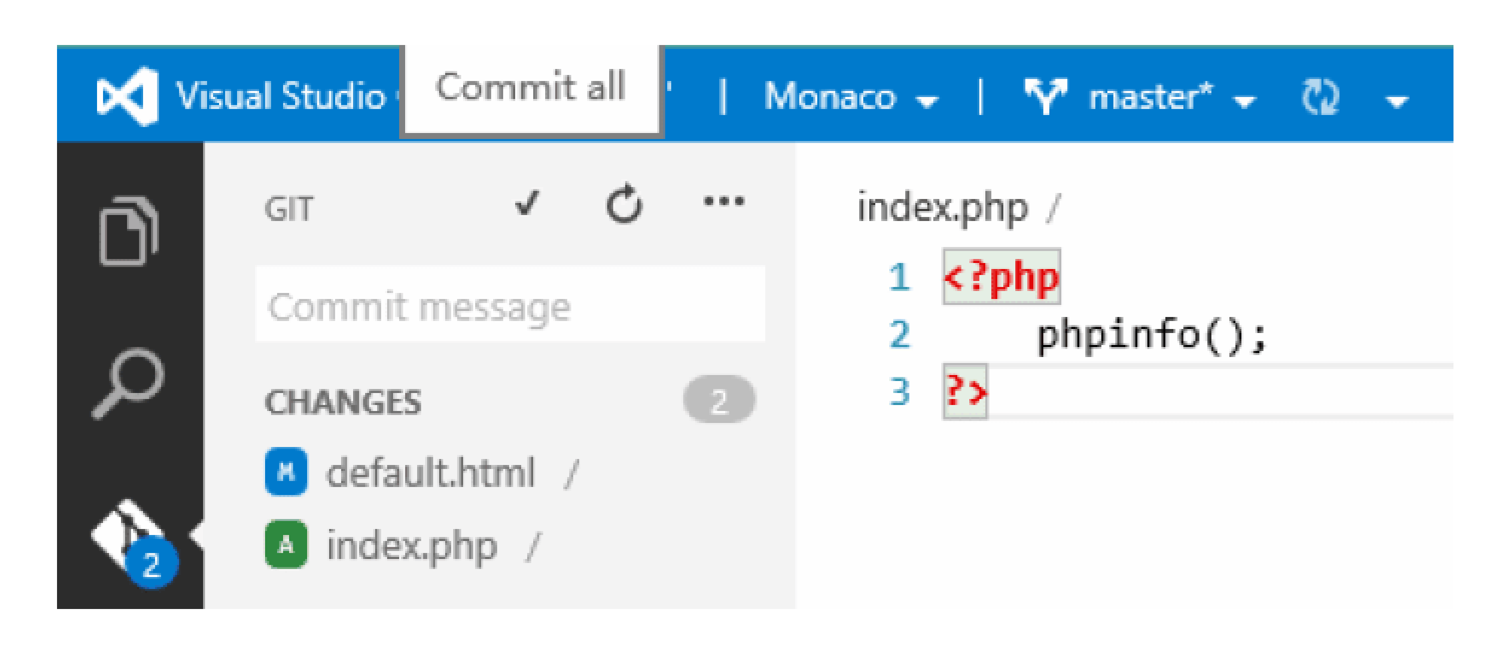


图 4-75　提交代码修改

（6）如图 4-76 所示，选中修改的文件后，Monaca 自动对比修改前后的文件并高亮显示修改的内容。

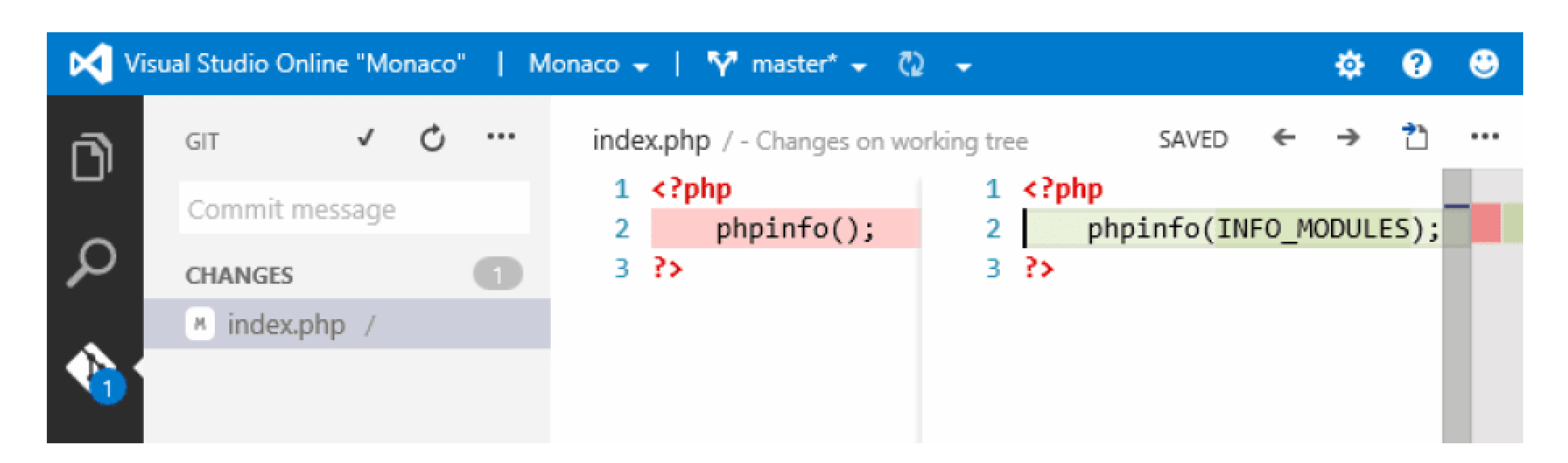


图 4-76　代码对比

（7）如图 4-77 所示，Monaco 在菜单栏显示本地 GIT 与 Visual Studio Online 中的区别。选择菜单中的相应命令可以将本地修改推送到 Visual Studio Online 或者从 Visual Studio Online 中下载最新的代码。

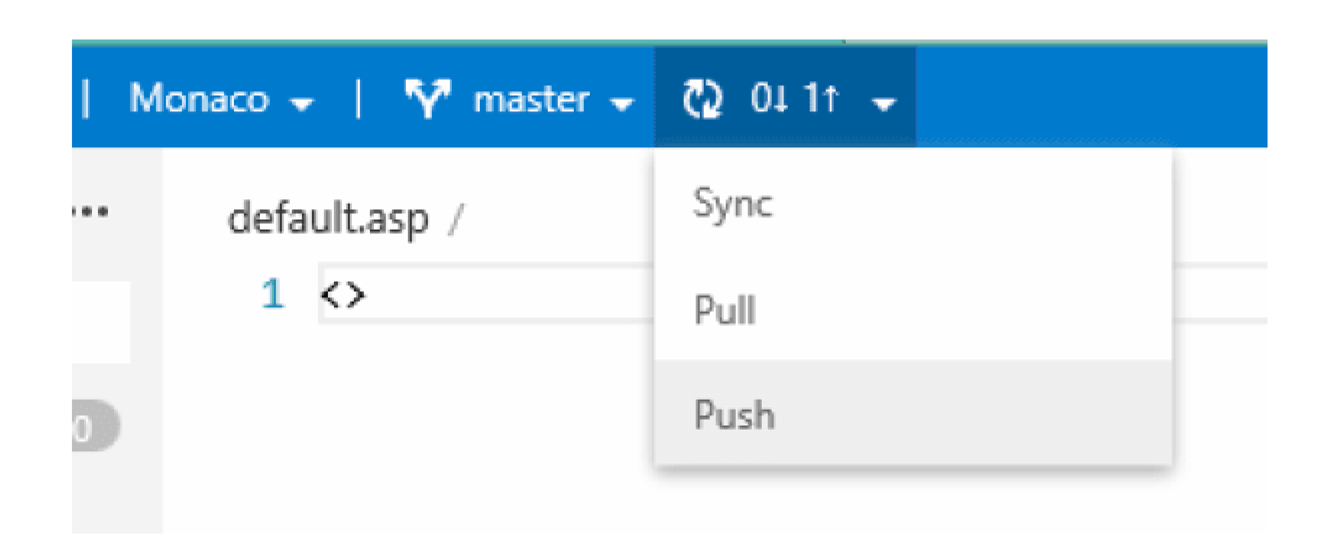


图 4-77　将改动推送到 Visual Studio Online

4.7.4　编辑源代码

Monaco 虽然没有完全媲美 Visual Studio 集成编辑环境的完整的代码编辑功能，但是仍然提供了非常友好、强大的编辑功能，包括智能提示、高亮代码段、快速注释代码等。

1. 智能编码提示

如图 4-78 所示，Monaco 智能提示可用方法、参数数目、类型等，可以提高开发效率。

```
server.js /
var http = require('http');
http.createServer(function (req, res) {
  res.writeHead(200, {'Content-Type': 'text/plain'});
  res.end('Hello World\n');
  console.lo
}).listen(pr   log (message?: any, ...optionalParams: any[]): void
```

图 4-78　智能代码提示

2. 智能 CSS 提示

如图 4-79 所示，将鼠标停留在 CSS 定义上，Monaco 可以显示引用该 CSS 定义的 HTML 元素。

```
<!DOCTYPE html><html><head>    <title>Microsoft Azure Website
100%;         }          #feature {          width: 960px;
font-family: "Segoe UI <element id="feature"> eight: normal;
       margin-top: 68px;             margin-left: 0px;
```

图 4-79　显示 CSS 提示

3. 智能错误提示

在编写代码时，Monaco 自动检测语法错误，并用红线标识错位位置。如图 4-80 所示，将鼠标停留在错误位置，Monaco 自动显示错误原因。

```
settings.php /
<?php
define('CACHE_LI Parse error: unexpected =
global $timeout = 100;
?>
```

图 4-80　语法错误提示

4. 快速注释代码

在 Monaco 中，选中任意代码，按 Ctrl+/快捷键，可以注释代码（或取消代码注释）。

5. 同时编辑多个文件

Monaco 允许同时显示和编辑多个文件。在很多情况下，该功能可以提高开发效率。比如，某个文件中引用了其他文件定义的 API，该功能可以直接查看 API 的定义，避免在多个文件中切换。

右键选择要打开的文件，然后选择 Open to the side 命令即可打开并行的新窗口编辑。在图 4-81 所示的例子中，index.php 引用了 settings.php，利用同时编辑功能无须在两个文件中切换即可方便地查看 CACHE_LIMITDE 的定义。

```
index.php                                SAVED  ×
<?php
    require_once('settings.php');
    echo CACHE_LIMIT;
?>
```

```
settings.php /
<?php
    define('CACHE_LIMIT', '100');

    global $timeout;
    $timeout = 100;
?>
```

图 4-81　同时编辑多个文件

6. 验证修改后的结果

在 Monaco 中，单击左侧导航栏的 Run 图标，即可运行网站，查看代码修改后的运行结果。

4.7.5　查看跟踪输出

前面介绍过可以通过跟踪功能查找软件错误。在 Monaco 中集成了实时跟踪信息查看功能。单击左侧导航栏的 Show output，页面中的跟踪信息会实时显示。

比如下面的代码利用 System.Diagnostics 输出跟踪信息：

```
protected void Page Load(object sender, EventArgs e)
{
    System.Diagnostics.Trace.WriteLine(Request.Headers["Host"]);
    System.Diagnostics.Trace.WriteLine(Request.Headers["X-ARR-LOG-ID"]);
    System.Diagnostics.Trace.WriteLine(Request.Url.Host);
}
```

当上面的代码执行时，在 Monaco 中可以看到对应的跟踪输出，如图 4-82 所示。

```
Output from http://wzhaotest.azurewebsites.net
[info] Output will appear here from the running application at
http://wzhaotest.azurewebsites.net
[info] The logging level is '16'. To change this, configure your application from
https://manage.windowsazure.com/#Workspaces/WebsiteExtension/Website/wzhaotest__5dca/configure
2014-11-08T10:10:02  PID[3452] Verbose     Request.Headers[Host]: wzhaotest.azurewebsites.net
2014-11-08T10:10:02  PID[3452] Verbose     X-ARR-LOG-ID : 6a52b863-5477-476a-9bfe-bb588a4861d5
2014-11-08T10:10:02  PID[3452] Verbose     Request.Url.Host: wzhaotest.azurewebsites.net
```

图 4-82　查看跟踪信息

4.7.6　命令行控制台

Monaco 包含一个命令行控制台，通过控制台可以直接执行 Monaco 自带的命令或者 Windows 命令。这些命令包括 npm、Nuget 和 Git 等。习惯命令行的开发人员可以通过这些命令管理应用。比如可以使用 npm 命令管理 Node 模块，或者通过 Nuget 命令管理.NET 扩展模块，还可以通过 Git 命令管理源代码。

4.8　参考文献与扩展阅读

1. ASP.NET 官方网站

ASP.NET 是一个免费开源的 Web 框架。通过 ASP.NET，可以开发 Web 网站、Web API 应用、移动站点以及利用 WebSocket 构建实时交互应用。

http://www.asp.net/

2. PHP 官方网站

PHP 是一个通用的脚本语言，尤其适用于开发 Web 应用。PHP 开放、灵活且容易使用，可以用于构建任何 Web 应用。

http://php.net/

3. Node.js

Node.js 是一个基于 Chrome JavaScript 引擎的平台，通过 Node.js，可以轻松创建快速、可扩展的网络应用。

http://nodejs.org/

4. IISNode

IISNode 是一个基于 IIS 的扩展模块。通过该模块可以将 Node.js 应用运行在 IIS 平台上。Node.js 应用在保留原有特点的同时又可以利用 IIS 丰富的功能。

https://github.com/tjanczuk/iisnode/

5. Load PHP Extension on Azure Websites

通过 PHP 扩展模块可以丰富 PHP 应用的功能。该文详细介绍了如何解决在 Azure 网站上配置 PHP 扩展模块时遇到的问题。

http://blogs.msdn.com/b/asiatech/archive/2013/12/30/why-my-php-extension-is-not-loaded-by-windows-azure-websites.aspx

6. 解决 PHP 性能问题

该文章介绍了如何解决部署在 Azure 网站上的 PHP 站点性能问题。

http://blogs.msdn.com/b/asiatech/archive/2013/11/15/azure-websites-find-php-performance-bottleneck.aspx

7. Debugging PHP using Windbg

该文章介绍了如何利用 Windows 调试器调试 PHP 性能问题。

8. Azure 网站中的 ASP.NET 无法加载程序集

无法加载程序集是 ASP.NET 应用部署到 Azure 网站中最常见的问题。该文章详细介绍了导致该问题的原因及解决方案。

http://blogs.msdn.com/b/asiatech/archive/2013/07/31/waws-could-not-load-file-or-assembly-msshrtmi.aspx

9. Node.js Tools for Visual Studio

Node.js Tools for Visual Studio（NTVS）是由微软公司设计、开发和支持的开源项目。通过 NTVS，可以在 Visual Studio 中编辑和调试 Node.js 应用。

https://nodejstools.codeplex.com/

10. File Structure on Azure

该文档描述了 Azure 网站中的文件目录结构，包括网站文件、日志文件和数据等。

https://github.com/projectkudu/kudu/wiki/File-structure-on-azure

11. Windows Azure Web Sites: How Application Strings and Connection Strings Work

在这篇文章中，Stefan 介绍了在 Azure 网站中如何设置应用选项和使用连接字符串。

http://azure.microsoft.com/blog/2013/07/17/windows-azure-web-sites-how-application-strings-and-connection-strings-work/

12.Visual Studio Online Monaco

该系列视频简要介绍了 Monaco 的各项功能，以及如何利用 Monaco 开发 ASP.NET、PHP 和 Node.js 应用。

http://channel9.msdn.com/Series/Visual-Studio-Online-Monaco

第 5 章　Azure 网站部署

Microsoft Azure 网站支持多种主流的部署方式。可以使用 Visual Studio 或者 Web Matrix 等集成开发工具将 ASP.NET、PHP 或者 Node.js 应用轻松部署到 Microsoft Azure 网站，也可以选择使用 FTP/FTPS 来部署应用，还可以使用 Web Deploy 命令行、PowerShell 等来自动化部署工作。

最令人兴奋的是 Microsoft Azure 网站支持多种源代码版本控制系统。无论使用哪种系统，都可以轻松地与 Microsoft Azure 网站集成。表 5-1 列出了 Microsoft Azure 网站支持的源代码版本控制系统。

表 5-1　Azure 网站支持的版本控制系统

系　　统	说　　明
Visual Studio Online	Visual Studio Online 提供了可用于软件管理、源代码管理、问题跟踪、生成和测试自动化等的完整解决方案
Git	将本地 Git 存储库发布到 Microsoft Azure 网站中的远程存储库
GitHub	GitHub 是与朋友、同事、同学和陌生人共享代码的最佳场所
Mercurial	Mercurial 是一种轻量级分布式版本控制系统，采用 Python 语言实现。支持将本地 Mercurial 存储库发布到 Microsoft Azure 网站
Dropbox	使用 Microsoft Azure 可快速同步和部署 Dropbox 文件夹中的代码
Bitbucket	Bitbucket 是一个针对 Git 和 Mercurial 分布式版本控制系统的托管网站。它包括问题跟踪程序、wiki 以及与许多常见服务（如 Basecamp、Flowdock 和 Twitter）的集成
CodePlex	CodePlex 是 Microsoft 的开放源项目托管网站。可以免费创建公共项目，与全球用户共享代码

本章首先介绍 Azure 网站的两种部署凭据。随后，介绍各种部署方式以及如何根据业务需要选择最合适的部署方式。

5.1　部署凭据

当部署应用到 Microsoft Azure 网站时，需要提供部署凭据（用户名和密码）来进行身份验证。Microsoft Azure 网站提供两种部署凭据：用户级部署凭据（又称为部署凭据）和站点级部署凭据（又称为发布配置文件凭据）。

5.1.1　用户级部署凭据

用户级部署凭据直接连接到一个 Microsoft Azure 订阅（与订阅 Microsoft Azure 的 Live

ID 相关联）而不是一个特定的网站。请注意，一个 Microsoft Azure 订阅可以有多个管理员/协同管理员，每个人有自己的一套的凭据。换句话讲，因为他们有不同的 Live ID，用户级凭据永远不可能在不同用户间共享。

可以在 Microsoft Azure 管理门户网站中按以下步骤设置部署凭据。

（1）登录到 Microsoft Azure 管理门户网站。

（2）选择任意一个站点。

（3）如图 5-1 所示，在“发布应用程序”下面，可以看到“重置部署凭据”。



图 5-1　重置部署凭据

或者，如图 5-2 所示，可以在任意网站的“仪表板”右侧的“速览”部分选择“重置部署凭据。”



图 5-2　通过仪表板重置部署凭据

5.1.2　站点级部署凭据（发布配置文件凭据）

站点级的部署凭据是由 Microsoft Azure 网站自动为每个网站生成的凭据。每个网站的凭据都不相同。站点级部署凭据保存在发布配置文件中，可以在 Microsoft Azure 管理门户网站中下载发布配置文件。

（1）登录到 Microsoft Azure 管理门户网站。

（2）选择要部署的站点，如图 5-1 所示，在快速开始页面单击“发布应用程序”下的“下载发布配置文件”。

或者在页面顶部导航栏单击“仪表板”，如图 5-3 所示，在“速览”下面，可以选择“下载发布配置文件”。

发布配置文件是一个 XML 文件，包含 Web Deploy 和 FTP 相关的配置和凭据。下面是一个发布配置文件的实例。发布配置文件主要包含发布应用需要的凭据、目标网站信息和链接的数据库资源等信息。站点级部署凭据中，用户名称的格式是“$[站点名称]“，比如

网站名称是 drumboy，那么用户名称就是$drumboy。密码是 60 位随机生成的字符。

速览

查看适用的外接程序

查看连接字符串

下载发布配置文件

图 5-3 下载发布配置文件

```
<publishData>
  <publishProfile
   profileName="drumboy - Web Deploy"
   publishMethod="MSDeploy"
   publishUrl="waws-prod-hk1-001.publish.azurewebsites.windows.net:443"
   msdeploySite="drumboy"
   userName="$drumboy"
   userPWD="wxfsaxJlhGe4NlkS0WiE37dJKacdLbE5RlrbqYHrJycymXbZ3Tcw8aiDuFF2"
   destinationAppUrl=http://drumboy.azurewebsites.net
   SQLServerDBConnectionString="Data Source=tcp:abc.database.windows.
   net,1433;
               Initial Catalog=abc;User ID=abc@p1yor6;Password=abc"
   mySQLDBConnectionString=""
   hostingProviderForumLink=""
   controlPanelLink=http://windows.azure.com
   targetDatabaseEngineType="sqlazuredatabase"
   targetServerVersion="Version100">
   <databases>
    <add
      name="drumboy"
      connectionString="Data Source=tcp:abc.database.windows.net,1433;
                  Initial Catalog=abc; User ID=abc@p1yor6;Password=abc"
      providerName="System.Data.SqlClient"
      type="Sql"
      targetDatabaseEngineType="sqlazuredatabase"
      targetServerVersion="Version100"/>
   </databases>
</publishProfile>
<publishProfile
 profileName="drumboy - FTP"
 publishMethod="FTP"
 publishUrl=ftp://waws-prod-hk1-001.ftp.azurewebsites.windows.net/site/wwwroot
 ftpPassiveMode="True"
 userName="drumboy\$drumboy"
```

```
    userPWD="wxfsaxJlhGe4NlkS0WiE37dacdmLLbE5RlrbqYHrJycymXbZ3Tcw8aiDuFF2"
    destinationAppUrl=http://drumboy.azurewebsites.net
    SQLServerDBConnectionString="Data Source=tcp:abc.database.windows.
    net,1433;
                    Initial Catalog=abc;User ID=abc@abc;Password=abc"
    mySQLDBConnectionString=""
    hostingProviderForumLink=""
    controlPanelLink=http://windows.azure.com
    targetDatabaseEngineType="sqlazuredatabase"
    targetServerVersion="Version100">
   <databases>
    <add name="drumboy"
     connectionString="Data Source=tcp:abc.database.windows.net,1433;
          Initial Catalog=abc;User ID=abc@abc;Password=abc"
     providerName="System.Data.SqlClient"
     type="Sql"
     targetDatabaseEngineType="sqlazuredatabase"
    targetServerVersion="Version100"/>
   </databases>
  </publishProfile>
</publishData>
```

不同于用户级凭据，站点级凭据很难记住。（当然，如果您记忆力超群，可以来试试！）通常，将发布配置文件导入集成开发环境，比如 Visual Studio 或者 Web Matrix，然后使用这些集成开发工具进行自动部署。

只要拥有发布配置文件，任何人都可以将网站部署到 Microsoft Azure 网站。比如，在开发阶段，将发布配置文件提供给开发人员进行测试并部署网站。网站上线后，开发人员仍然拥有部署网站的权限。基于安全考虑，希望取消开发人员部署网站的权限。这时，可以重置发布配置文件凭据。此后，之前下载的发布配置文件中的用户密码将立即失效。开发人员如果希望继续发布文件到 Microsoft Azure 网站，需要下载新的发布配置文件。

通过 Microsoft Azure 管理门户重置发布配置文件凭据的步骤如下：

（1）登录到 Microsoft Azure 管理门户网站。

（2）选择要重置发布凭据的站点。

（3）单击顶部导航栏的“仪表板”，如图 5-4 所示，在“速览”下面，单击“重置您的发布配置文件凭据”。

速览

查看适用的外接程序
查看连接字符串
下载发布配置文件
重置部署凭据
重置您的发布配置文件凭据

图 5-4　重置发布配置文件凭据

5.1.3 如何选择部署凭据

两种部署凭据都可以用来部署应用到 Microsoft Azure 网站。通常，可以选择使用用户级别的部署凭据，因为它更容易记忆。但是，有些情况下可能选择站点级部署凭据。比如，希望某个开发团队只能拥有部署项目到某个网站的权限，可以将站点级部署凭据提供给该开发团队。

如果使用用户级部署凭据，用户名称为<sitename>\<you deploy user>，比如 drumboy\DeployUsr。

如果使用站点级部署凭据，用户名称为<sitename>\$<sitename>，比如 drumboy\$drumboy。

5.2 使用 FTP 部署网站

无论用什么样的集成开发环境，都可以使用 FTP 将网站文件部署到 Microsoft Azure 网站。FTP 部署方式简单易用，可以使用任何 FTP 客户端，包括浏览器（如 Internet Explorer）、功能齐全的免费工具（如 FileZilla）和集成开发环境（比如 Visual Studio、WebMatrix、Eclipse 等）都支持 FTP；甚至可以通过 FTP 命令行编写脚本来自动化 FTP 的部署工作。另外，Microsoft Azure 网站支持更多的安全 FTPS 协议。

虽然 FTP 部署方式简单易用，只需要将网站文件复制到网站的 site/wwwroot 目录下即可。但是它有很多限制。比如，FTP 部署不能自动处理配置/修改数据库连接字符串等常见的部署任务。同时，许多 FTP 工具不比较源和目标文件，不能自动跳过没有改变的文件。因此，有时候即使只有很少的修改，也需要复制所有文件。对于大型网站，尤其是文件数量较多的网站，FTP 部署所耗时间相对其他方式要长得多。

下面列出了 Microsoft Azure 网站的具体文件目录结构。

```
LogFiles
    Application
        <pid>-<ticks>-<instance>.txt        //应用程序诊断日志
    DetailedError
        ErrorPage####.htm                   //详细的错误消息
    Git
        trace
            trace.xml                       //Git 部署时产生的跟踪信息
            <instance>-<guid>.txt           //Kudu 相关的信息
        deployment
            <instance>-<guid>.txt           //与部署相关的信息
    http
        RawLogs
            <logfile>.log                   //IIS 日志（每 60s 更新一次）
```

```
    W3SVC#####
        fr####.xml                    //IIS 失败请求跟踪
        freb.xsl
site
    wwwroot
        hello.htm                     //网站内容
    repository                        //网站源代码管理存储库
        .git
            HEAD, index and other git files
    deployments
        [commit id 1]
            log.xml                   //部署日志，与管理门户显示的信息类似
            status.xml                //部署状态（成功/失败）
            manifest                  //部署的文件列表
        [commit id 2]
            ...
    .ssh
        config                        //配置
        id_rsa                        //私有密钥
        known hosts                   //已知的主机列表
Data
    Jobs
        Triggered
            MyTriggeredJob1
                20131112101559
                    output.log        //WebJob 输出
                    error.log         //WebJob 错误信息
                    status            //WebJob 执行状态

        Continuous
            MyContinuousJob1
                job.log               //WebJob 日志
                status                //WebJob 执行状态
```

5.3 Web Deploy

Web Deploy 提供了一个将 ASP.NET Web 程序部署到远程服务器的自动化方式。相对于 FTP 方式，Web Deploy 非常聪明。在部署过程中，它会对比本地项目和远程服务器的文件，它只复制修改的文件或者新的文件。所以如果只是对一个大项目做了一点改动并重新发布的话，只有修改过的文件会被复制过去。Web Deploy 不会重新复制没有被修改的文件。这就让重新部署/更新一个站点快很多，特别是那些有很多静态内容和大图片的项目。

Web Deploy 不仅允许发布文件，而且可以维护数据库结构/数据，执行数据库变更脚

本，设置文件访问权限等。在部署过程中 Web Deploy 可以修改/定制 web.config 文件，部署数据库更新。

Web Deploy 提供了丰富多样的 Provider（适配器组件）用于部署特定的数据，比如数据库、文件、注册表和配置等。同时，允许用户开发自己的 Provider 用于处理自己应用的特殊情景。

Visual Studio 集成了 Web Deploy 的功能，可以在 Visual Studio 直接通过 Web Deploy 部署 Web 应用。也可以在 Visual Studio 中创建部署包，可以通过命令行和脚本将部署包部署到 Microsoft Azure 网站。

此外，Visual Studio Express、Web Matrix 等免费开发工具也集成了 Web Deploy 的功能。IIS 管理器也集成了 Web Deploy 功能。

关于 Web Deploy 的功能介绍，请参考下面的文档：

http://www.iis.net/learn/publish/using-web-deploy/introduction-to-web-deploy

5.3.1 Visual Studio 中使用 Web Deploy 发布网站

Visual Studio 集成了 Web Deploy 和 Microsoft Azure 订阅管理功能。可以轻松地从 Visual Studio 中发布 Web 应用到 Microsoft Azure。下面以 Visual Studio 2013 为例演示如何使用 Web Deploy 将网站发布到 Microsoft Azure。

（1）在 Visual Studio 2013 中，打开 Web 应用。

（2）在“解决方案资源管理器”中，右击要发布的项目，选择“发布”，打开“发布 Web”对话框。

（3）如图 5-5 所示，Visual Studio 默认提供了 3 个发布目标。



图 5-5 选择发布目标

① Microsoft Azure 网站。

选择 Microsoft Azure 网站，如图 5-6 所示，Visual Studio 要求登录到 Microsoft Azure。登录后，Visual Studio 会获取网站列表，可以选择将 Web 应用部署到一个现有的网站，或者新建一个网站。Visual Studio 会自动获取网站的发布配置文件，并将其导入用户的工程项目中。

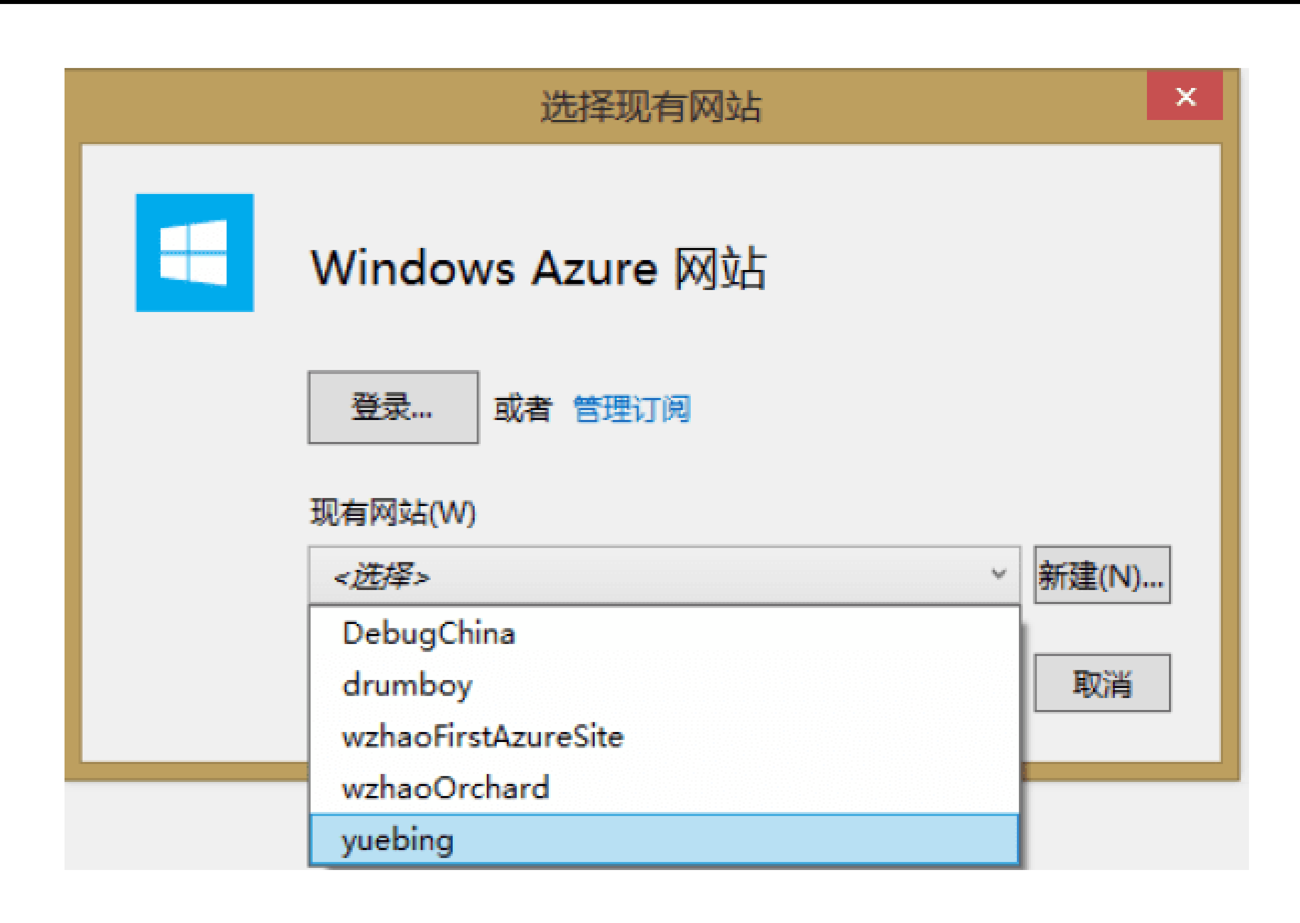


图 5-6　发布到 Microsoft Azure 网站

② 导入发布配置文件。

如果是开发人员，没有访问 Azure 订阅的权限，那么可以要求网站管理员提供发布配置文件。前面提到过，发布配置文件是一个 XML 文件，包含 Web Deploy 和 FTP 相关的配置和凭据。通过该选项，可以将发布配置文件导入到 Web 工程项目。

③ 自定义。

通常，发布 Web 应用到 Microsoft Azure 网站并不需要自定义如何发布网站。因为自定义所需要提供的信息都包含在发布配置文件中，直接导入发布配置文件即可。

（4）验证连接。

选择要发布的网站后，进入“连接”页面，此时可以单击“验证连接”按钮来验证网络连接。如图 5-7 所示，出现绿色对号表示验证成功，单击“下一步”按钮。

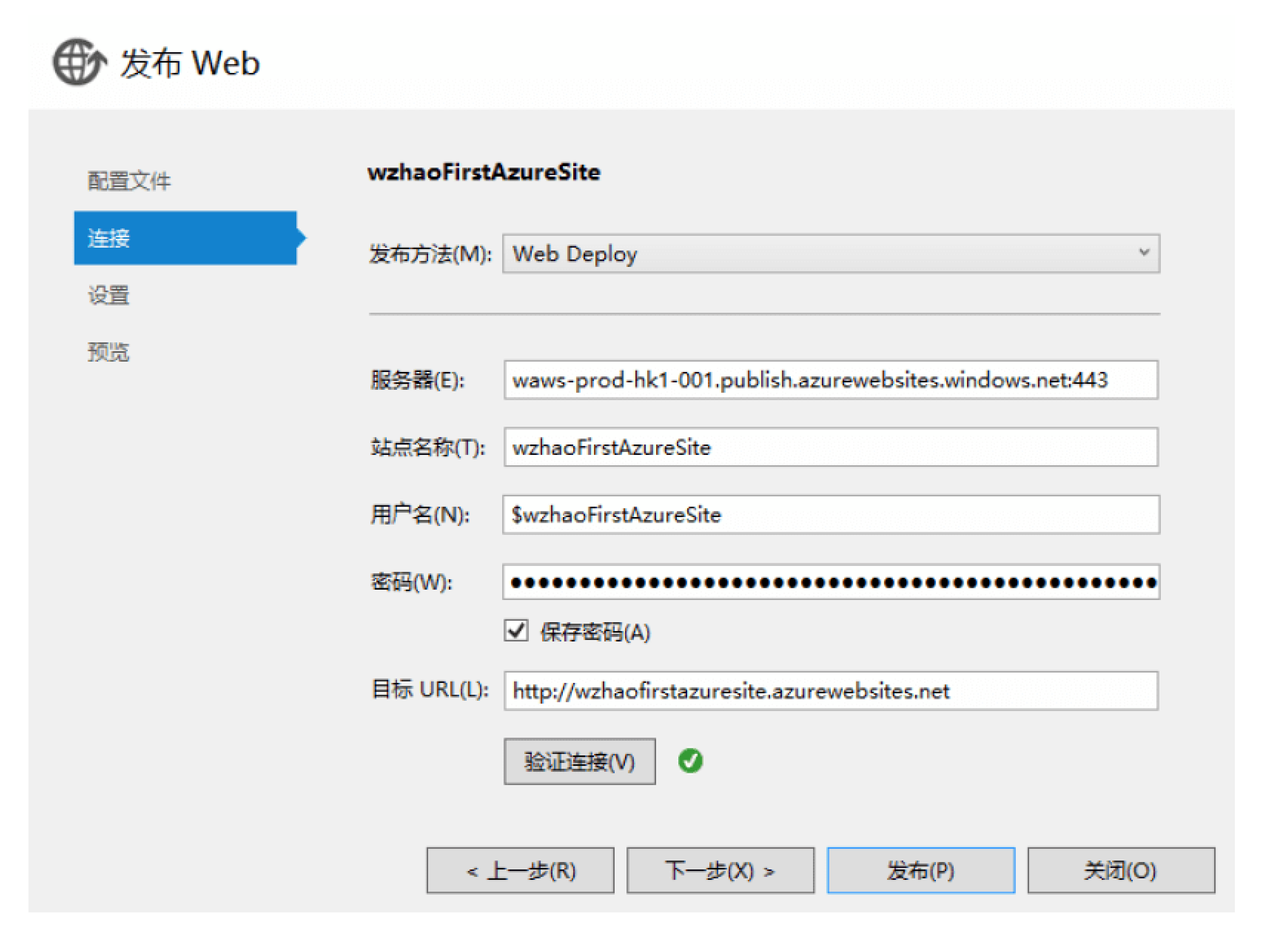


图 5-7　验证连接

（5）设置发布选项。

如图 5-8 所示，可以在此步骤设置发布选项，包括：

- Release 版本或者 debug 版本。
- 文件发布选项。
- 数据库选项。



图 5-8 设置发布选项

（6）发布预览。

在预览窗口，单击“发布预览”按钮，Web Deploy 会比较本地和 Microsoft Azure 网站的文件，并生成需要更新的文件列表。Web Deploy 只发布需要更新的文件。

（7）单击“发布”按钮，将 Web 应用发布到 Microsoft Azure 网站。

在 Visual Studio 的输出窗口中，会看到下面的输出信息:

```
1>------ 已启动生成: 项目: MyFirstAzureSite, 配置: Release Any CPU ------
1>MyFirstAzureSite ->
    C:\TestCode\WebSitesBook\MyFirstSite\
    MyFirstAzureSite\bin\MyFirstAzureSite.dll
2>------ 发布已启动: 项目: MyFirstAzureSite, 配置: Release Any CPU ------
2>已使用 C:\TestCode\WebSitesBook\MyFirstSite\MyFirstAzureSite\Web.
Release.config 将 Web.config 转换为 obj\Release\TransformWebConfig\
transformed\Web.config。
2>已将自动 ConnectionString obj\Release\TransformWebConfig\transformed\
Web.config 转换为 obj\Release\CSAutoParameterize\transformed\Web.config。
2>正在将所有文件都复制到以下临时位置以进行打包/发布:
2>obj\Release\Package\PackageTmp。
2>启动 Web Deploy 以将应用程序/包发布到 https://waws-prod-hk1-001.
publish.azurewebsites.windows.net/msdeploy.axd?site=wzhaoFirstAzureSite
...
2>正在添加路径的 ACL (wzhaoFirstAzureSite)
2>正在添加路径的 ACL (wzhaoFirstAzureSite)
2>正在更新文件(wzhaoFirstAzureSite\bin\MyFirstAzureSite.dll)
2>正在更新文件(wzhaoFirstAzureSite\Web.config)
2>正在添加路径的 ACL (wzhaoFirstAzureSite)
2>正在添加路径的 ACL (wzhaoFirstAzureSite)
```

```
2>发布成功。
2>站点已成功发布 http://wzhaofirstazuresite.azurewebsites.net/
========== 生成:  成功 1 个，失败 0 个，最新 0 个，跳过 0 个 ==========
========== 发布:  成功 1 个，失败 0 个，跳过 0 个 ==========
```

5.3.2　Visual Studio 部署 MVC 应用（后台使用数据库）

Web Deploy 的一个优点就是支持数据库部署。下面以 Contoso University 为例，演示如何使用 Web Deploy 部署后台使用数据库的应用。Contoso University 是一个基于 ASP.NET MVC5 的 Web 应用程序，它使用了 Entity Framework 6。可以从下面的站点下载源代码：

http://code.msdn.microsoft.com/ASPNET-MVC-Application-b01a9fe8

（1）登录到 Microsoft Azure 管理门户网站。

（2）单击左下角的“新建”按钮，选择“计算”→“网站”。

（3）选择“自定义创建”，如图 5-9 所示，提供如下信息：

- URL：输入要创建的网站名称。该名称必须是未被注册的名称，这里以 ContosoEDU 为例。
- WEB 宿主计划：可以选择已有的 Web 宿主计划，或者选择创建新的 Web 宿主计划。这里选择已有的计划。
- 区域：选择一个数据中心，选择原则是尽量靠近网站的客户。
- 数据库：创建新的 SQL 数据库。
- 指定数据库连接字符串名称。



图 5-9　自定义创建网站

（4）如图 5-10 所示，指定数据库的名称和区域，单击“完成”按钮。

（5）在管理门户网站，下载 ContosoEDU 网站的发布配置文件。

（6）运行 Visual Studio 2013，打开 Contoso University 项目，该项目使用 SQL EXPRESS LOCAL DB，默认的连接字符串为



图 5-10　指定数据库设置

```
connectionString="Data Source=(LocalDb)\v11.0;
    Initial Catalog=ContosoUniversity2;
    Integrated Security=SSPI;"
    providerName="System.Data.SqlClient" />
```

（7）在解决方案资源管理器中，右击 ContosoUniversity 项目，选择“发布”。

（8）如图 5-11 所示，单击“导入”，在“导入发布设置”对话框中指定 ContosoEDU 的发布配置文件。

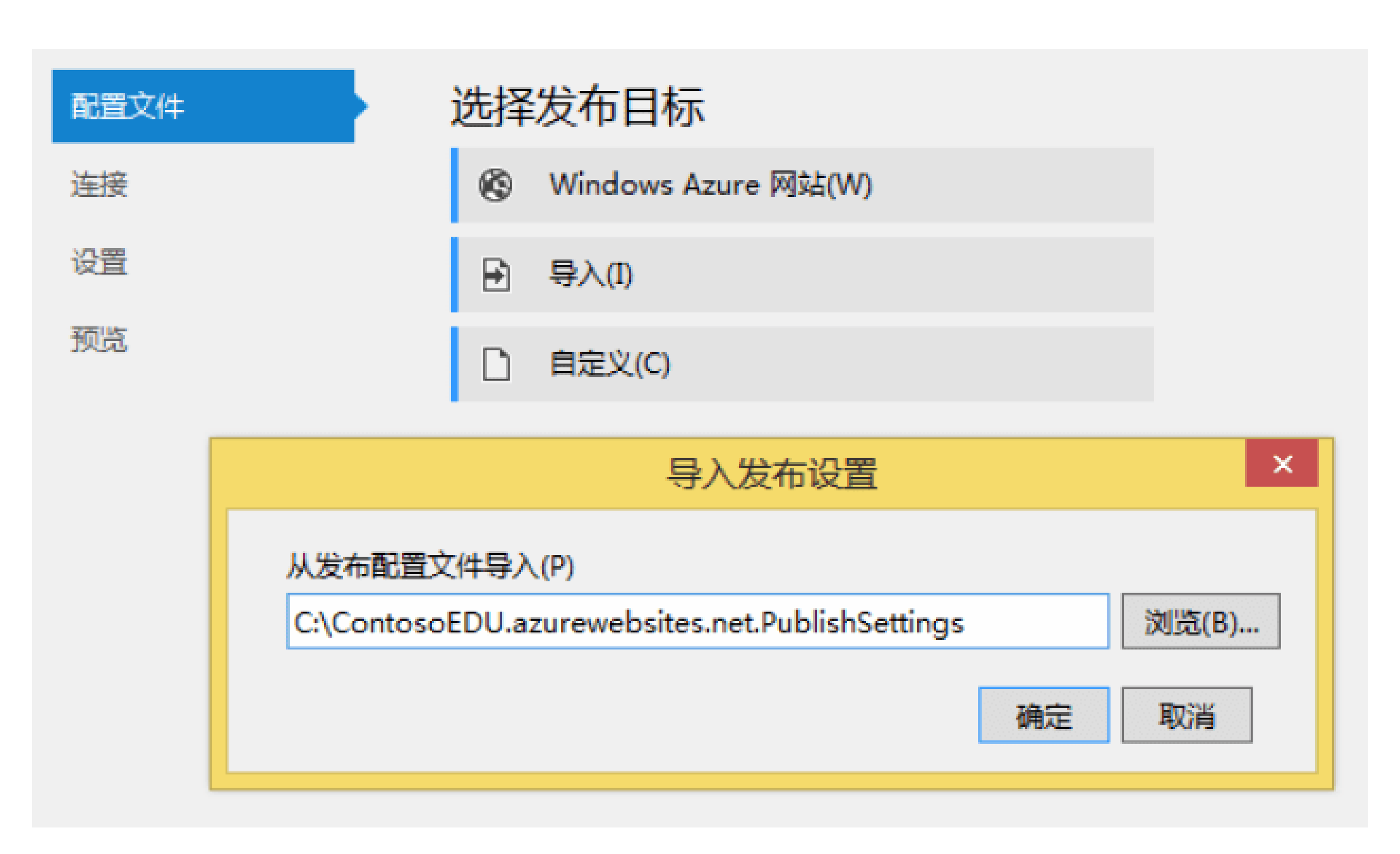


图 5-11　导入发布设置

（9）如图 5-12 所示，在“设置”对话框，可以看到 Web Deploy 自动检测到使用的 SQL Azure 的连接字符串，并用该连接字符串替换本地使用的连接字符串。

选择下面的两个选项：

- 在运行时使用此连接字符串（更新目标 web.config）。
- 执行 Code First 迁移（在应用程序启动时运行）。



图 5-12　数据库设置

（10）单击“发布”按钮将网站部署到 Microsoft Azure 网站。

5.3.3　Web Deploy 命令行

除了与 Visual Studio 集成外，Web Deploy 同时提供了强大的命令行工具 MSDeploy.exe。有些情景下，可能更希望使用命令行部署应用，而不是从 Visual Studio 或者 Web Matrix 等开发工具中直接部署。

- 如果使用自动化脚本在部署过程中进行一些自动化、定制的操作，那么可以将 msdeploy.exe 部署命令集成到部署脚本中。
- 如果只有网站的内容文件，没有网站的源代码，比如一个现有的网站，那么可以通过 MSDeploy.exe 命令行将网站部署到 Microsoft Azure 网站。

下面的命令将 c:\myBlogSite\下的文件同步到 mySite.azurewebsites.net 网站：

```
msdeploy.exe
    -verb:sync
    -source:contentPath="c:\myBlogSite"
    -dest:
        contentPath=’mySite,
ComputerName="https://waws-prod-hk1-001.publish.azurewebsites.windows.net:
443/msdeploy.axd?site=mySite",
        UserName='$myBlogOnAzure,
        Password='asdfghjk;lkjhgfd',
          AuthType='Basic'
```

-verb：指定 Web Deploy 进行的操作，sync 表示同步操作，将源文件同步到目标网站。

-source：指定同步的源文件。可以用 contentPath 指定目录，或者使用 package 指定一个压缩包，或者指定一个本地运行的网站。

-dest：指定同步的目标网站。ComputerName 对应于发布配置文件中的 PublishUrl。具体格式为 https://<*PublishUrl*>/msdeploy.axd?site=<*sitename*>。

比如，在发布配置文件中，

```
PublishUrl="waws-prod-hk1-001.publish.azurewebsites.windows.net:443"
```

站点名称为 mySite，那么 ComputerName 需要指定为

https://waws-prod-hk1-001.publish.azurewebsites.windows.net:443/msdeploy.axd?site=mySite"

UserName 和 Password 在发布配置文件中都可以找到。Microsoft Azure 网站的 Web Deploy 只接受 Basic 认证方法，所以 AuthType 要指定为 Basic。

5.3.3.1 WAWSDeploy

如上所见，使用 msdeploy.exe 部署网站时，需要指定的参数非常复杂。首先，需要下载发布配置文件，然后将需要的信息从发布配置文件中提取出来。最后，指定 msdeploy.exe 命令行所需要的参数。为了简化这个过程，David Ebbo 发布了一个小工具可以自动指定 msdeploy.exe 的命令行参数，从而简化了 msdeploy.exe 的使用。可以从 David Ebbo 的博客下载该工具：

http://blog.davidebbo.com/2014/03/WAWSDeploy.html

该工具的使用非常简单。

（1）登录到 Microsoft Azure 管理门户网站。

（2）下载目标站点的发布配置文件。

（3）运行下面的命令行将本地文件夹的内容部署到 Microsoft Azure 网站：

```
WAWSDeploy c:\FolderToDeploy MyAzureSite.PublishSettings
```

下面是一个运行实例：

```
WAWSDeploy.exe C:\msdeploypackage\MyFirstAzureSite c:\publishFile\mysite.
publishSettings
```

5.3.3.2 Web Deploy 3.6 Beta 版本更新

Web Deploy 3.6 Beta 版本中添加了两个新功能：

（1）支持代理服务器。Web Deploy 功能强大，但是目前版本的 Web Deploy 不支持代理服务器。如果必须使用代理服务器，那么可以尝试 Web Deploy 3.6 Beta 版本。

（2）支持指定发布配置文件简化命令行参数，命令行简化如下：

```
msdeploy.exe
-verb:sync
-source:contentPath=c:\siteName\wwwroot
-dest:contentPath=siteName,publishsettings=c:\siteName.PublishSettings
```

5.4 Git

Git 诞生于 2005 年，是一个免费的、分布式的版本控制工具，可以有效、高速地支持从很小到非常大的项目版本管理。Git 简单易用，速度飞快，适合管理大项目，它还有

着令人难以置信的非线性分支管理系统，可以应付各种复杂的项目开发需求。Git 已经成为占有率第一的版本控制系统。

Microsoft Azure 网站支持从 Git 代码库中将 Web 应用直接发布到 Azure 网站。

5.4.1　Project Kudu

Microsoft Azure Git 部署是通过 Kudu 来支持的。Kudu 是一个开源的项目，它的项目网站是：

http://github.com/projectkudu/kudu

每个 Azure 网站都有一个与之关联的 SCM 网站，该网站运行 Kudu。如果 Azure 网站地址是 http://contoso.azurewebsites.net，那么 Azure 网站所对应的 SCM 网站的地址就是 https://contoso.scm.azurewebsites.net。

注意事项：

（1）SCM 网站只允许通过 HTTPS 访问。

（2）SCM 网站不受自有域名的影响。比如网站配置了自有域名 www.mysite.com，客户通过 http://www.mysite.com 来访问网站。但是 SCM 网站并不受影响，只能通过 https://mysite.scm.azurewebsites.net 来访问 SCM 站点。

（3）SCM 站点要求认证，可以使用 Azure 订阅账户或者发布凭据登录。

5.4.2　使用 Git 发布 Web 应用到 Microsoft Azure 网站

如果已经使用 Git 来管理 Web 应用开发，可以直接从第四步开始。

1．安装 Git

Git 支持多个平台，在不同平台上安装步骤不尽相同。具体安装步骤请参考下面的文档：

http://git-scm.com/book/en/Getting-Started-Installing-Git

2．创建一个本地代码库

按照以下步骤来创建一个新的代码仓库。

（1）创建一个目录来包含 Git 仓库和网站的文件，比如 GitSite。

（2）打开命令行，切换到 GitSite 目录下。运行下面的命令来初始化一个新的 Git 仓库：

```
C:\TestCode\GitSite>git init
Initialized empty Git repository in C:/TestCode/GitSite/.git/
```

3．添加网站文件

Microsoft Azure 网站支持多种语言，下面以 PHP 为例进行介绍。

（1）在 Git 仓库（即先前创建的 GitSite 目录）下创建一个名为 index.php 的新文件，包含如下内容：

```
<html>
<head>
    <title>Hello PHP</title>
</head>

<body>
   <h1>PHP Site V1</h1>
   <?php
        phpinfo();
    ?>
</body>
</html>
```

（2）打开命令行，切换到 GitSite 目录下。运行下面的命令将 index.php 文件添加到代码仓库：

```
git add index.php
```

（3）运行下面的命令将 index.php 提交到代码仓库：

```
C:\TestCode\GitSite>git commit -m "initial version"
[master (root-commit) 99d5c03] initial version
 1 file changed, 13 insertions(+)
 create mode 100644 index.php
```

4．启用 Microsoft Azure 网站代码仓库

（1）登录到 Microsoft Azure 管理门户网站。

（2）在左边的导航栏选择“网站”，然后选择要启用代码仓库的网站。

（3）如图 5-13 所示，在快速开始页面的“集成源代码管理”下，单击“从源代码管理设置部署”。

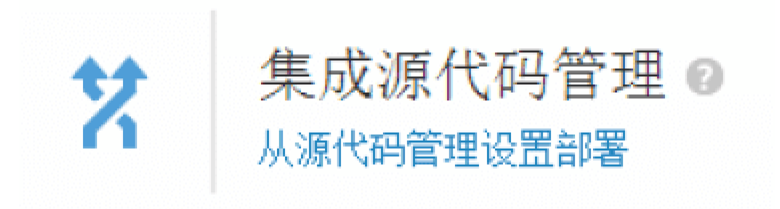


图 5-13　集成源代码管理

或者在“仪表板”页面，单击“速览”下面的“从源代码管理设置部署”。

（4）在“设置部署”对话框中，选择“本地 Git 存储库”，单击“下一步”按钮。

稍等片刻，Git 存储库即可创建成功。如图 5-14 所示，创建成功后，可以看到存储库的 URL。

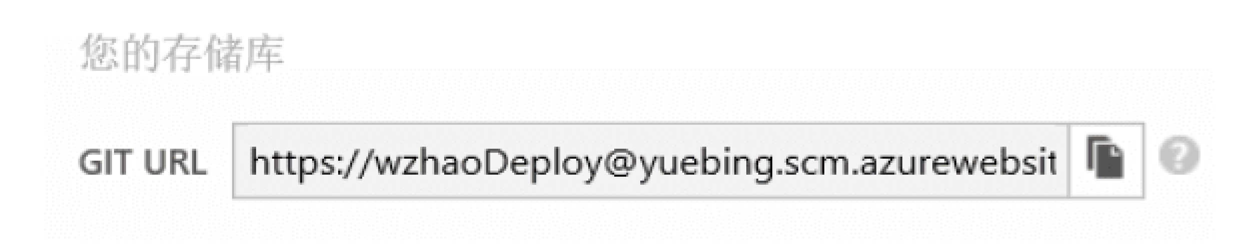


图 5-14　Git URL

Git URL 的格式是 https://<deployuser>@<sitename>.scm.azurewebsites.net:443/<sitename>.git。

5. 部署网站

打开命令行，切换到 GitSite 目录下，然后运行下面的命令将文件推送到 Azure 网站：

```
git remote add azure https://wzhaoDeploy@yuebing.scm.azurewebsites.net:443
/yuebing.git
git push azure master
```

推送时，需要提供部署密码，下面是 git push azure master 的输出示例：

```
C:\TestCode\GitSite>git push azure master
Password for 'https://wzhaodeploy@yuebing.scm.azurewebsites.net:443':
Counting objects: 3, done.
Delta compression using up to 4 threads.
Compressing objects: 100% (2/2), done.
Writing objects: 100% (3/3), 362 bytes | 0 bytes/s, done.
Total 3 (delta 0), reused 0 (delta 0)
remote: Updating branch 'master'.
remote: Updating submodules.
remote: Preparing deployment for commit id '99d5c0366e'.
remote: Generating deployment script.
remote: Generating deployment script for Web Site
remote: Running deployment command...
remote: Handling Basic Web Site deployment.
remote: KuduSync.NET from: 'D:\home\site\repository' to: 'D:\home\site\
wwwroot'
remote: Copying file: 'index.php'
remote: Finished successfully.
remote: Deployment successful.
To https://wzhaodeploy@yuebing.scm.azurewebsites.net:443/yuebing.git
 * [new branch]      master -> master
```

6. 更新代码并推送至 Azure 网站

（1）打开 index.php，将文件修改为如下内容（版本修改为 V2）：

```
<html>
<head>
    <title>Hello PHP</title>
</head>

<body>
   <h1>PHP Site V2</h1>
   <?php
```

```
        phpinfo();
        ?>
    </body>
    </html>
```

（2）回到命令行，运行下面的命令将修改添加到代码仓库：

```
git add index.php
```

（3）运行下面的命令提交修改：

```
git commit -m "version 2"
```

（4）将更新推送到 Azure 站点：

```
git push azure master
```

7. 查看网站部署历史记录

（1）登录到 Microsoft Azure 管理门户网站。
（2）在左边的导航栏选择“网站”，然后选择通过 Git 部署的网站。
（3）单击顶部的“部署”，打开“部署”页面。
（4）如图 5-15 所示，可以看到网站的部署历史记录。



图 5-15　部署历史记录

（5）如果当前版本有问题，可以选择之前的旧版本，重新部署。

5.4.3　将现有网站克隆到本地 Git 存储库

有些情况下，可能希望将网站已有的内容克隆到本地的 Git 存储库。比如，通过 Microsoft Azure 网站的 Web 应用库安装了 WordPress 网站，希望将该网站纳入本地 Git 进行版本管理，这时需要将该网站克隆到本地。

下面的步骤演示了如何将网站克隆到本地，并将本地更新推送到 Azure 网站。

（1）启用网站存储库。启用网站存储库的步骤与 5.4.2 节第 4 步相同。

（2）打开命令行，切换到要保存本地代码的目录。

（3）运行下面的命令，需要提供部署密码。如前所示，可以从管理门户网站获取 Git URL。

```
git clone <Git URL>
```

（4）命令运行结束后，可以看到一个以网站命名的文件夹。该文件夹包含网站/site/wwwroot 目录下的所有文件。从命令行切换到该文件夹。

（5）打开任意网站文件并修改，比如 index.php。

（6）依次运行下列命令。

① 回到命令行，运行下面的命令将修改添加到代码仓库：

```
git add index.php
```

② 运行下面的命令提交修改：

```
git commit -m "version 2"
```

③ 将更新推送到 Azure 站点：

```
git push azure master
```

5.5　从 Visual Studio Online 部署

Visual Studio Online 是微软公司运行的基于 TFS 系统的在线版本管理系统。本节主要介绍如何从 Visual Studio Online 中部署一个 ASP.NET+SQL 的应用到 Azure 网站。

5.5.1　将 Visual Studio Online 中的项目部署到 Azure 网站

（1）首先需要一个有效的 Microsoft Azure 订阅。

根据订阅信息，选择正确的地址登录到 Microsoft Azure 管理门户网站。

Microsoft Azure 全球管理门户网站：

https://manage.windowsazure.com

Microsoft Azure 中国区管理网站（由世纪互联公司运营）：

https://manage.windowsazure.cn

（2）需要一个 Visual Studio Online 的账户。本例演示将 Visual Studio Online 中的一个 ASP.NET+SQL 的应用部署到 Azure 网站。

（3）单击左下角的“新建”按钮，选择“计算”→“网站”。将会看到 3 个选项：“快速创建”、“自定义创建”和“从库中创建”。这里选择“自定义创建”。

（4）出现“创建网站”对话框，如图 5-16 所示，指定下列信息：

① 网站名称。

② 选择订阅。

③ 选择数据中心，网站将被建立在指定的数据中心。尽量选择靠近网站客户的数据中心，以降低网络延迟。

④ 指定数据库选项：

- 不需要数据库。
- 创建新的 SQL 数据库。
- 创建新的 MySQL 数据库。
- 使用已有 SQL/MySQL 数据库。

⑤ 选择“从源代码管理发布”。



图 5-16　自定义创建网站

（5）单击“下一步”按钮，指定数据库设置，如图 5-17 所示。为了获得更好的性能，应该将网站和数据库创建在同一个数据中心。



图 5-17　指定数据库设置

（6）单击“下一步”按钮，配置高级数据库设置，如图 5-18 所示。请根据应用需要选择合适的选项。



图 5-18　高级数据库设置

（7）单击“下一步”按钮，选择代码管理选项，如图 5-19 所示，请选择 Visual Studio Online。



图 5-19　选择代码管理系统

（8）单击“下一步”按钮，进入授权连接，如图 5-20 所示。输入 Visual Studio Online 的 URL，并单击“立即授权”。

（9）授权成功后，会自动转到最后一步：选择要部署的存储库，如图 5-21 所示。用户在 Visual Studio Online 里面所有的存储库都会被列出，选择想要关联的存储库，然后单击“完成”。



图 5-20　授权连接



图 5-21　选择源代码存储库

（10）等待几十秒钟，网站即创建完成。

此时，用户已经拥有一个 Azure 网站，并将该网站与 Visual Studio Online 中的网站应用关联在一起。此后，每一次签入代码（Check-in code）后，用户所作的修改都会自动被部署到 Azure 网站。

5.5.2　从 Visual Studio 中部署代码更新

Azure 网站与 Visual Studio Online 集成后，当开发人签入代码时，Visual Studio Online 自动编译代码并将编译后的项目部署到 Azure 网站。下面的步骤具体演示了该功能。

（1）在 Azure 管理门户网站的命令栏，单击 Visual Studio 图标。Visual Studio 会自动打开将与 Azure 网站关联的 Visual Studio Online 网站应用。

也可以在 Visual Studio 中单击“团队”，选择“连接到 Team Foundation Server”连接到 Visual Studio Online 账户。

还可以登录到 Visual Studio Online 网站，在网站中选择 Open in Visual Studio。

（2）在 Visual Studio 中修改代码，然后签入代码。

（3）回到 Azure 管理门户网站，如图 5-22 所示，打开“部署”页面。会看到应用正在部署。



图 5-22　网站部署

（4）部署成功后，打开网站，即可看到更新后的网页。

5.5.3　Visual Studio Online 集成 Azure 网站工作原理

Visual Studio Online 与 Azure 网站集成后，Visual Studio Online 拥有一个 Azure 订阅的管理证书，因此 Visual Studio Online 可以通过 Azure 管理 REST API 管理 Azure 网站的资源。同时 Visual Studio Online 的项目中会创建一个新的生成定义（Build Definition），该生成定义中包含了部署到 Azure 网站的步骤。

当开发人员签入代码后：

（1）Visual Studio Online 中项目的生成定义会被触发。

（2）生成定义触发后，编译项目代码。

（3）Visual Studio Online 通过管理证书调用 Azure 管理 REST API 下载 Azure 网站的部署文件，该文件包含将项目部署到 Azure 网站所需的信息。

（4）Visual Studio Online 通过 Web Deploy 将编译结果部署到 Azure 网站。

（5）如果 Visual Studio Online 中的项目是通过 Git 进行版本管理的，Visual Studio Online 通过 Git 将修改部署到 Azure 网站。

5.6　从 GitHub 中部署

GitHub 是一个基于 Web 的系统，它使用 Git 版本控制系统托管项目源代码库。GitHub

免费托管公开源代码的开源项目。对于企业来讲，可以付费托管私有库。

作为开源代码库以及版本控制系统，GitHub 目前拥有 140 多万开发者用户。随着越来越多的应用程序转移到了云上，GitHub 已经成为管理软件开发以及发现已有代码的首选方法。

Azure 网站作为一个开放平台，无缝支持 GitHub。托管在 GitHub 上的 Web 应用可以轻松部署到 Azure 网站。本节讨论如何将 Azure 网站与 GitHub 集成起来。

5.6.1　集成 Azure 网站与 GitHub 存储库

本节演示如何集成 Azure 网站与 GitHub 存储库。

（1）登录到 Microsoft Azure 管理门户网站。

（2）在左边的导航栏，选择“网站”，然后选择要集成 GitHub 的网站。

（3）如图 5-23 所示，在“快速开始”页面的“集成源代码管理”下，单击“从源代码管理设置部署”。

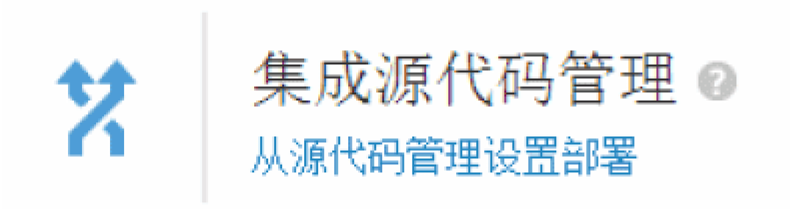


图 5-23　集成源代码管理

或者在“仪表板”页面，单击“速览”下面的“从源代码管理设置部署”。

（4）在“设置部署”对话框中，选择 GitHub，单击“下一步”按钮。

（5）在授权页面，输入 GitHub 的用户名和密码登录。

（6）授权成功后，如图 5-24 所示，选择要部署的存储库，然后单击“完成”按钮。

选择要部署的存储库

存储库名称

WAWSUsingAzureStorage

要部署的分支

master

图 5-24　选择 GitHub 存储库

（7）在顶部导航栏单击“部署”，打开部署页面。如图 5-25 所示，GitHub 中选择的存储卡已经部署到 Azure 网站。



图 5-25　GitHub 部署历史记录

5.6.2　将 GitHub 中的项目部署到 Azure 网站

本节演示如何在 Visual Studio 2013 中签入代码到 GitHub 并将代码更新部署到 Azure 网站。

注意：Visual Studio 2013 内建支持 Git，如果使用 Visual Studio 2012 或者 2010，则需要安装 Visual Studio Tools for Git 工具。

将 GitHub 存储库克隆到本地的步骤如下：

（1）在 Visual Studio 2013 中，打开团队资源管理器，在“本地 Git 存储库”选项下，选择“克隆”。

（2）如图 5-26 所示，输入 GitHub 存储库的 URL，将 GitHub 的存储库克隆到本地。



图 5-26　将 GitHub 存储库克隆到本地

（3）在 Visual Studio 中编辑代码，然后提交修改。

（4）提交代码改动后，如图 5-27 所示，修改被提交到本地。单击同步链接。



图 5-27　提交代码修改

（5）如图 5-28 所示，在团队资源管理器中，单击“推送”链接，将修改提交到 GitHub 中。单击推送链接后，需要输入 GitHub 的用户名和密码进行认证。



图 5-28　将修改推送到 GitHub

（6）回到 Microsoft Azure 管理门户网站，在集成 GitHub 的网站的“部署”页面，如图 5-29 所示，可以看到新的修改已经自动同步到了 Azure 网站。



图 5-29 GitHub 部署历史记录

5.6.3 GitHub 与 Azure 网站集成工作原理

GitHub 与 Azure 网站集成后，Azure 网站在 GitHub 存储库中创建了一个 Webhook。当特定事件发生时（比如代码提交），GitHub 通过 Webhook 通知外部服务。Azure 网站创建的 Webhook 告诉 GitHub 当代码提交事件发生时调用下面的 URL 通知 Azure 网站：

https://<sitename>.scm.azurewebsites.net/deploy

该 URL 指向 SCM 网站。5.4 节详细介绍了 SCM 网站。

SCM 网站收到 GitHub 的请求后，利用 Git 从 GitHub 中获取最新的代码，然后在本地进行编译并部署。

5.7 阶 段 部 署

Microsoft Azure 网站标准模式下运行的网站支持分阶段的部署工作。可以为生产网站创建开发或阶段部署（staging）网站。可以在生产网站和阶段部署网站之间交换，这种交换可以实现零停机时间部署。阶段部署的流程如下：

（1）将网站更新部署到阶段部署网站（staging site）。

（2）在阶段部署网站上验证更新。

（3）验证通过后，交换生产网站与阶段部署网站。

（4）此时，生产网站变为阶段部署网站，阶段部署网站升级为生产网站。

（5）监控生产网站，如果发现问题，可以通过再次交换进行回滚，将旧版本的生产网站重新切换为当前生产网站。

图 5-30 描述了阶段部署的流程。

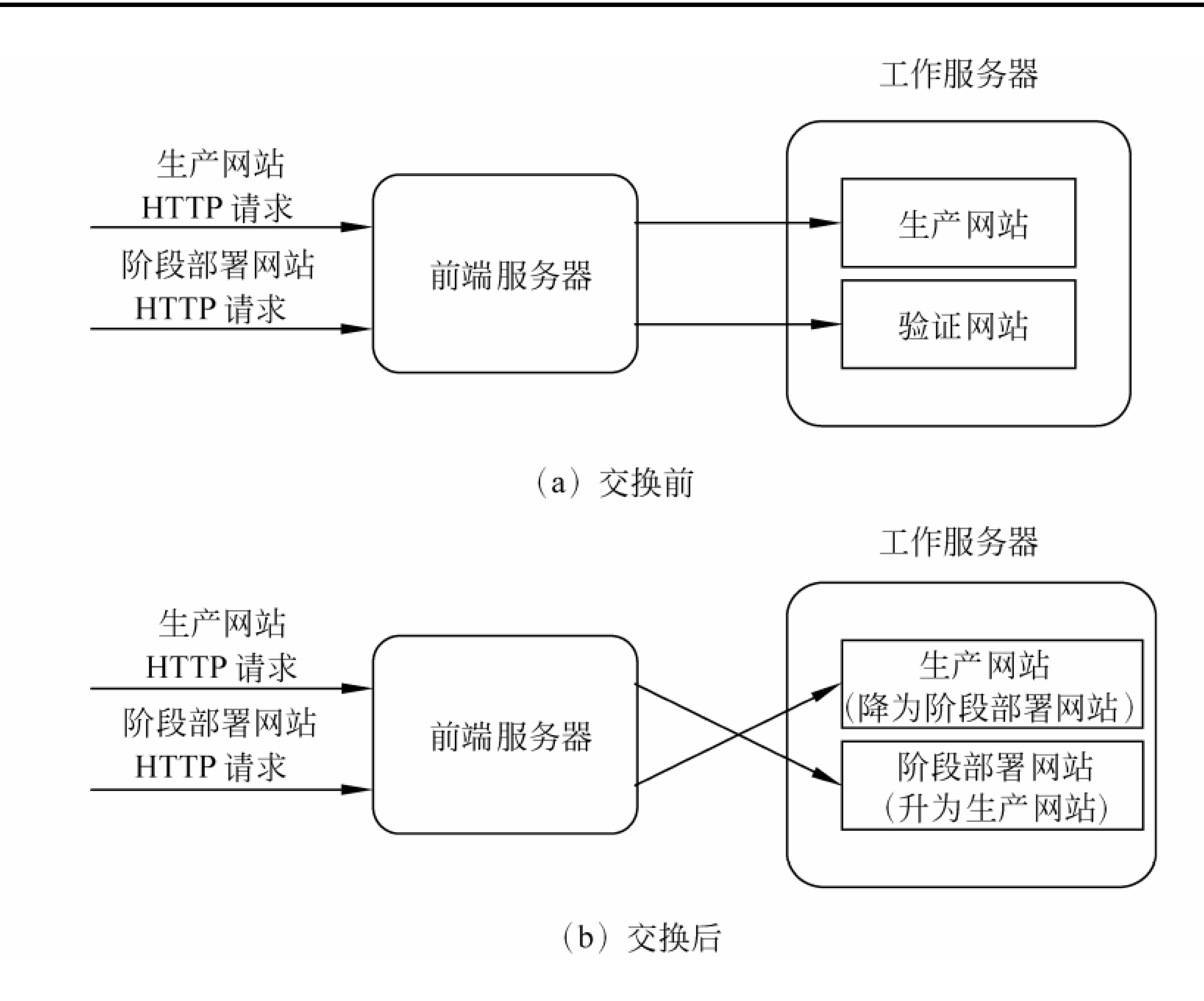


图 5-30　阶段部署示意图

分阶段的部署方案具有如下优点：

（1）功能验证。在正式部署之前将内容或配置部署到一个阶段部署网站后，可以验证部署后的网站。验证通过后，可以通过网站交换将阶段部署网站升级为生产网站。

（2）构建和整合网站内容。可以逐步将内容更新添加到阶段部署网站，当更新完成后，通过网站交换将阶段部署网站升级为生产网站，完成更新。

（3）回滚生产网站。如果发现新版本的更改与预期不符，或者有其他严重问题，可以进行网站交换，将原来的内容回滚为生产环境。

Microsoft Azure 网站在将网站交换为生产网站之前自动对网站进行预热（warm up），避免冷启动造成的性能暂时下降。在交换过程中，HTTP 请求重定向是无缝的，不会因为网站交换行为导致 HTTP 请求被丢弃的现象。换句话讲，交换是零停机时间。目前，每个标准模式的网站可以创建 4 个阶段部署网站。

5.7.1　阶段部署与网站配置

当创建一个部署网站时，部署网站的配置默认从生产网站克隆。所有部署网站的配置是可以编辑的。当生产网站与部署网站交换时，下面的配置将交换：

- 常规设置。
- 连接字符串。
- 处理器映射。
- 监控和诊断设置。

下面的配置在网站交换时不会改变，这些配置永远与当前的生产网站绑定在一起。

- 发布端点。
- 自定义域名。

- SSL 证书和绑定。
- 缩放设置。

5.7.2 使用阶段部署实现零停机部署

下面具体演示如何使用阶段部署功能实现网站零停机部署。

1. 添加阶段部署网站

（1）登录到 Microsoft Azure 管理门户网站，选择要启用阶段部署的网站（必须是标准模式的网站）。

（2）在“快速开始”页面，单击“发布应用程序”下的“添加新的部署槽”，如图 5-31 所示。

图 5-31 从“快速开始”页面添加新的部署槽

或者在“仪表板”页面，单击“速览”下面的“添加新的部署槽”，如图 5-32 所示。

图 5-32 从“仪表板”页面添加新的部署槽

（3）在“添加新的部署槽”对话框中输入名称，如图 5-33 所示。

图 5-33 指定部署槽名称

（4）创建成功后，如图 5-34 所示，可以看到阶段部署网站。

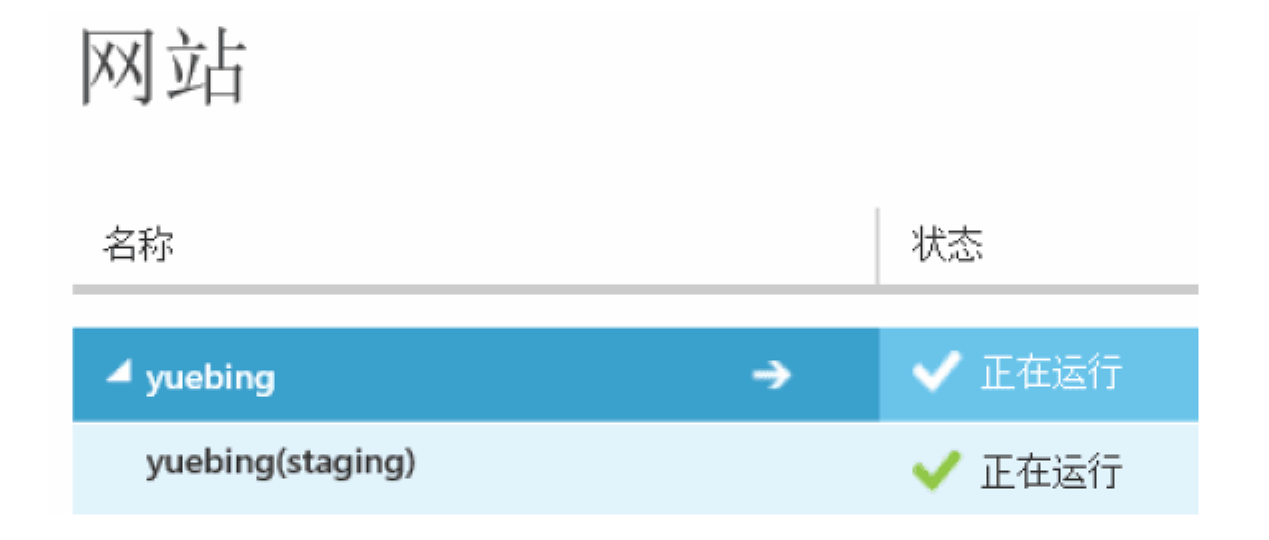


图 5-34　阶段部署网站

2．上传页面

在生产网站（yuebing）下，上传一个页面 index.php，包含如下内容：

```
<html>
  <head>
    <title>Yuebing</title>
  </head>
  <body>
    <?php
      echo "<h1>Hello yuebing V1!</h1>";
      $host = $_SERVER['HTTP_HOST'];
      echo "<p>Running on ";
      echo $host;
      echo "</p>";
    ?>
  </body>
</html>
```

此时，访问网站，如图 5-35 所示，看到 V1 网站运行在 yuebing 网站。

Hello yuebing V1!

Running on yuebing.azurewebsites.net

图 5-35　网站初始版本

3．将网站更新部署到阶段部署网站

将 index.php 修改为 V2，内容如下，并部署到阶段部署网站。

```
<html>
  <head>
    <title>Yuebing</title>
```

```
    </head>
    <body>
      <?php
        echo "<h1>Hello yuebing V2!</h1>";
        $host = $_SERVER['HTTP_HOST'];
        echo "<p>Running on ";
        echo $host;
        echo "</p>";
      ?>
    </body>
</html>
```

如图 5-36 所示，访问 yueming-staging 网站，可以确认网站已经升级为 V2。

Hello yuebing V2!

Running on yuebing-staging.azurewebsites.net

图 5-36　阶段部署网站升级到版本 2

4．站点交换

经过测试后，可以将阶段部署网站升级为生产网站。如图 5-37 所示，在 Azure 管理门户网站，选择要交换的网站，单击命令栏的“交换”按钮。



图 5-37　交换网站

如图 5-38 所示，在“交换部署”对话框中，选择源网站与目标网站，单击“完成”按钮。



图 5-38　选择源网站与目标网站

5．验证交换

网站经完成交换后，生产网站变为阶段部署网站（V1），阶段部署网站升级为生产网

站。此时，访问 http://yuebing.azurewebsites.net 网站，看到生产网站已经升级为 V2，如图 5-39 所示。

Hello yuebing V2!

Running on yuebing.azurewebsites.net

图 5-39　生产网站升级到版本 2

6. 网站回滚

如果发现新部署的版本有问题，可以再进行一次网站交换回滚到原来的生产网站。

关于阶段部署网站有以下几个注意事项：

（1）部署插槽仅适用于在标准模式下的网站。如果想将网站降到免费模式、共享模式或者基本模式，必须首先删除部署网站。要删除部署网站，步骤如下：

① 登录到管理门户网站。

② 打开要删除的部署网站。如图 5-40 所示，在底部命令栏，可以选择删除部署网站或者同时删除部署网站和生产网站。



图 5-40　删除部署槽

（2）打算切换到生产环境的部署网站应按照生产环境的需要进行配置。

（3）默认情况下，部署网站将指向与生产网站相同的数据库。但是，可以通过配置部署网站数据库连接字符串将部署网站指向一个备用数据库。在交换之前，需要修改部署网站的数据库连接字符串指向生产环境的数据库。

（4）缩放不适用于部署网站，而仅适用于生产网站。

（5）阶段部署网站不支持链接的资源管理。

（6）如果需要，即使启用了阶段部署，仍然可以根据需要直接将应用发布到生产网站。

（7）部署网站与生产网站共享相同的资源，并运行在同一台虚拟机上。如果对部署网站进行压力测试，生产网站将会受到影响。

5.7.3　使用 PowerShell 管理阶段部署

Microsoft Azure PowerShell 命令行提供了用于管理阶段部署的命令。下面的示例假定已经有一个名为 yuebing 的网站，该网站在 East Asia 数据中心。

1. New-AzureWebsite

可以通过使用 New-AzureWebsite 命令，并同时指定网站和部署网站的名称创建一个部署网站。部署网站和生产网站必须在同一数据中心。如下面的示例所示：

```
New-AzureWebsite yuebing -Slot staging -Location "East Asia"
```

该命令将为 yuebing 网站创建一个名为 yuebing-staging 的部署网站。如果 yuebing 网站不存在，该命令将产生一个名为 yuebing 的网站，同时为 yuebing 网站创建一个名为 yuebing-staging 的部署网站。

2. Switch-AzureWebsiteSlot

Switch-AzureWebsiteSlot 命令执行交换操作，将生产网站与部署网站交换。

3. Remove-AzureWebsite

如果部署网站不再需要，可以通过使用 Remove-AzureWebsite 命令将其删除。下面的命令删除 yuebing-staging 网站：

```
Remove-azurewebsite -Name yuebing -Slot staging
```

5.7.4 使用 X-CLI 管理阶段部署

Microsoft Azure 跨平台命令行提供了用于管理阶段部署的命令。下面的示例假定已经有一个名为 yuebing 的网站，该网站在 East Asia 数据中心。

1. azure site create

可以通过使用 azure site create 命令，并同时指定网站和部署网站的名称创建一个部署网站。部署网站和生产网站必须在同一数据中心。如下面的示例所示：

```
azure site create yuebing --slot staging
```

该命令将为 yuebing 网站创建一个名为 yuebing-staging 的部署网站。如果 yuebing 网站不存在，该命令将产生一个名为 yuebing 的网站，同时为 yuebing 网站创建一个名为 yuebing-staging 的部署网站。

2. azure site swap

azure site swap 命令执行网站交换操作，将生产网站与部署网站交换。

3. azure site delete

如果部署网站不再需要，可以通过使用 azure site delete 命令将其删除。下面的命令删除 yuebing-staging 网站：

```
azure site delete yuebing --slot staging
```

5.8　在生产环境中进行测试

通过阶段部署，可以在生产环境和部署环境之间无缝切换。开发人员首先将新版本的应用部署到部署环境进行验证。验证通过后，通过将部署环境与生产环境互换的方式将新版本应用部署到生产环境。假如新版本在生产环境中发现问题，可以通过部署槽切换将旧的稳定版本切换回生产环境。

阶段部署可以实现不宕机部署，并降低部署风险。在发现问题后可以及时切换回之前的稳定版本。但是，阶段部署仍然是基于传统的测试方法，主要由测试人员进行各种测试。现代互联网应用要求更短的发布周期，而且，随着各种新的交互技术的应用，在测试环境中模拟客户的行为愈发困难。对于大规模 Web 应用而言，创造一个能够完全模拟生产环境的测试和开发环境几乎是不可能完成的任务。

为了解决该难题，在生产环境中进行测试（Test in Production，TiP）是最近几年非常流行的测试方法。TiP 的核心思想就是通过在生产环境中测试，最小化产品风险，加快发布节奏。TiP 通过暴露新代码给有限用户，减少缺陷可能带来的负面影响，通过在产品中暴露这些新代码，可以快速获得这些新代码的反馈，这些反馈来自真实的用户，而不是少量的测试人员和有限的测试用例。另外，一旦发现新代码有严重的缺陷，可以快速修复这些缺陷，发布新版本或者回滚到老版本。

通常，在生产环境中测试 Web 应用需要一个路由引擎和分析反馈回路。Azure 网站提供了一个路由平台进行生产测试。它的基本思想是将一部分真实客户的网络请求分发到指定的阶段部署网站。如图 5-41 所示，在开始阶段可以将少量客户请求（比如将 10%的客户请求）转发到阶段部署网站。如果测试结果符合预期，则逐步将更多的客户请求转发到阶段部署网站，直到将所有的用户请求转发至阶段部署网站。

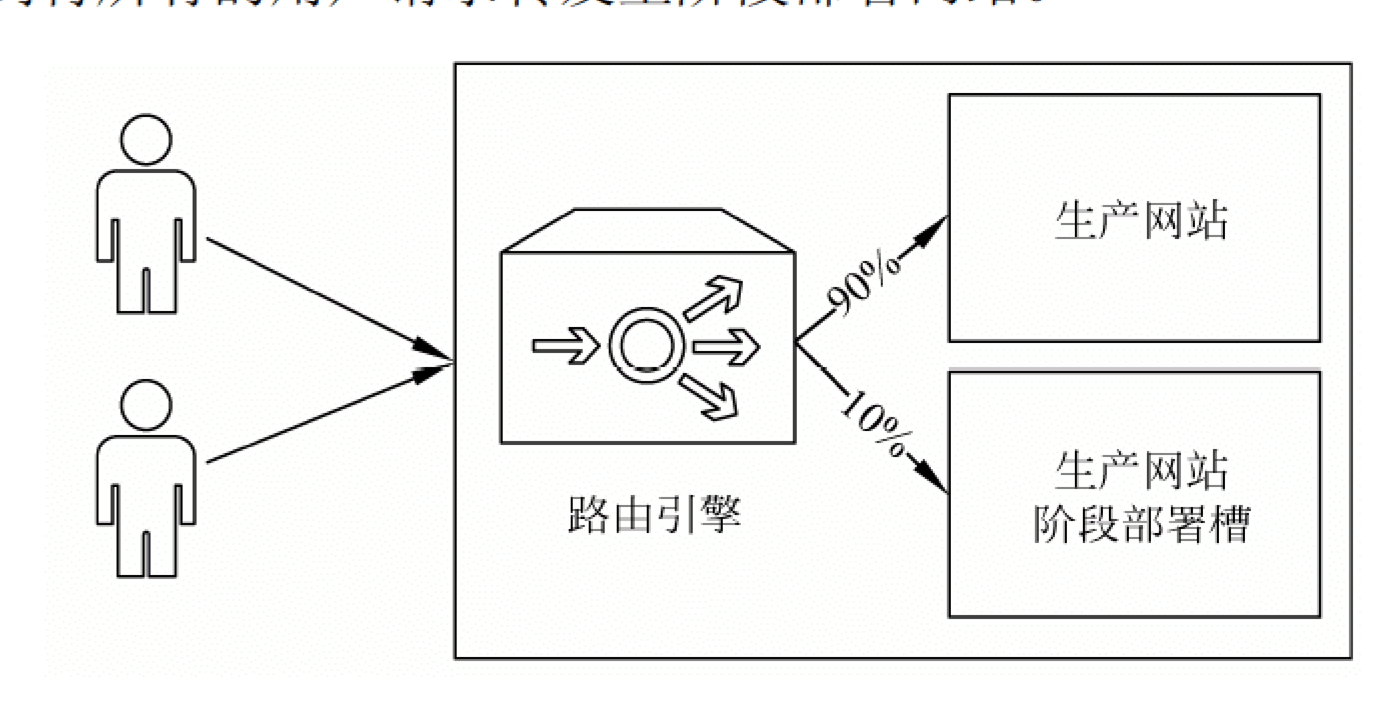


图 5-41　在生产环境中测试示意图

本节讨论如何配置 Azure 网站在生产环境中进行测试。

5.8.1　创建网站

只有标准模式的网站可以使用在生产环境中测试的功能。首先需要一个标准模式的网

站，并创建至少一个部署槽。有关创建部署槽的步骤，请参考 5.7 节。

在本例中，首先创建一个标准模式的网站，网站名称为 TestInProduction。之后，在该网站上添加一个部署槽，名称为 staging，部署网站名称为 TestInProduction-Staging。

5.8.2 部署测试代码

如 4.7 节中所述，使用 Monaco 在 TestInProduction 网站的根目录下创建一个新的文件，命名为 default.cshtml。该文件包含如下代码：

```
<!DOCTYPE html>
<html lang="en">
  <head>
    <title>Azure Websites AppSettings</title>
  </head>
<body>
   <h1>Production</h1>
   <div>
      <p>Welcome to : @Environment.
                        GetEnvironmentVariable("WEBSITE HOSTNAME")</p>
       <p>Hosting Plan Mode: @Environment.
                        GetEnvironmentVariable("WEBSITE COMPUTE MODE")</p>
    </div>
</body>
</html>
```

如图 5-42 所示，该文件在浏览器中显示网站的名称和模式。

Production

Welcome to : testinproduction.azurewebsites.net

Hosting Plan Mode: Dedicated

图 5-42　示例网站

5.8.3 配置在生产环境中测试功能

当前的管理门户网站不支持配置在生产环境中测试功能，但是可以通过 PowerShell/CLI 或者 Beta 版本的新门户网站配置在生产环境中测试功能。

5.8.3.1 使用 PowerShell 配置在生产环境中测试功能

使用 PowerShell 需要首先安装 Microsoft Azure PowerShell，关于如何安装 Microsoft Azure PowerShell，请参考 3.4 节。

下面的 PowerShell 脚本将 10%的客户请求转发到 TestInProduction 的 staging 部署槽，

剩余的 90%请求被转发到 TestInProduction 网站。

```
#使用 Azure 账号登录
Add-AzureAccount

#如果您的账号关联了多个订阅，该命令设置默认的订阅
Select-AzureSubscription -SubscriptionName myDefaultSub

#指定网站名称
$sitename = "TestInProduction"

#创建请求转发规则
$rule = New-Object
      Microsoft.WindowsAzure.Commands.Utilities.Websites.Services.
      WebEntities.RampUpRule
$rule.ActionHostName = "testinproduction-staging.azurewebsites.net"
$rule.ReroutePercentage = 10
$rule.Name = "staging"

$list = New-Object System.Collections.Generic.List
      [Microsoft.WindowsAzure.Commands.Utilities.Websites.Services.
      WebEntities.RampUpRule]
$list.Add($rule)

#设置路由转发规则
Set-AzureWebsite -Name $sitename -Slot Production -RoutingRules $list
```

5.8.3.2　通过 Beta 门户网站配置在生产环境中测试功能

通过 Azure Beta 管理门户网站（仅限于全球 Azure 环境，不支持中国区 Azure 环境），可以直接配置在生产环境中测试的功能。

（1）登录到 Azure Beta 管理门户网站 https://portal.azure.com/。

（2）单击左侧命令栏的“浏览”，然后选择“网站”。

（3）在网站列表中选择要配置的网站，在本例中是 TestInProduction。

（4）在 TestInProductioN 网站的配置页面，定位到“生产测试”部分。图 5-43 展示了运行上面的 PowerShell 命令后的结果，10%的客户请求将被转发至 staging 部署槽。

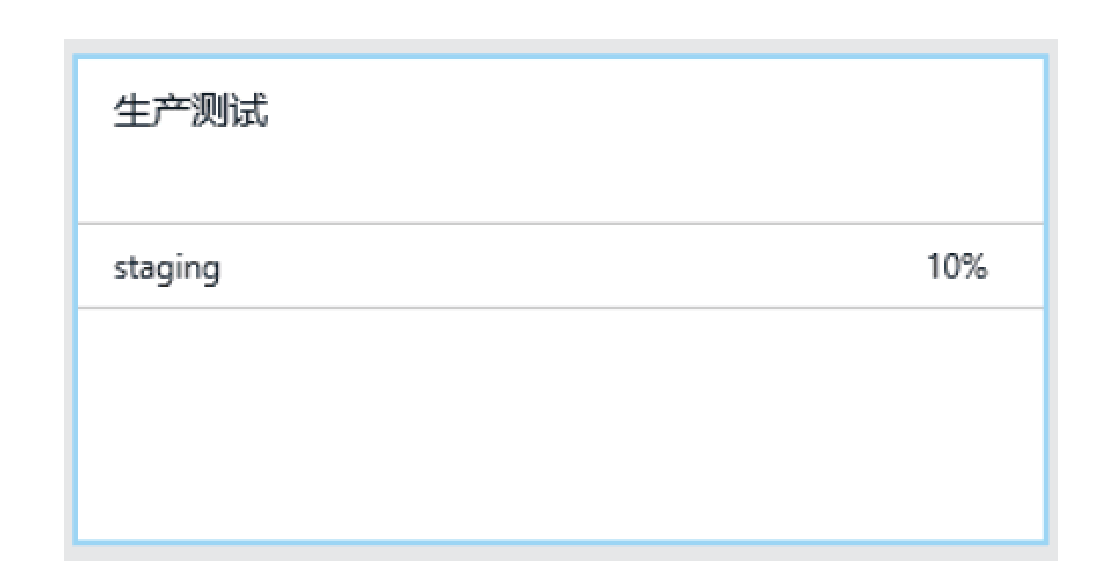


图 5-43　配置生产测试

（5）单击“生产测试”，如图 5-44 所示，可以修改/配置请求分配比例。或者可以选择将请求分配到其他的部署槽。比如，40%的请求分配至 staging 部署槽，10%的请求分配至 Dev 部署槽，50%的请求分配至 production 网站。

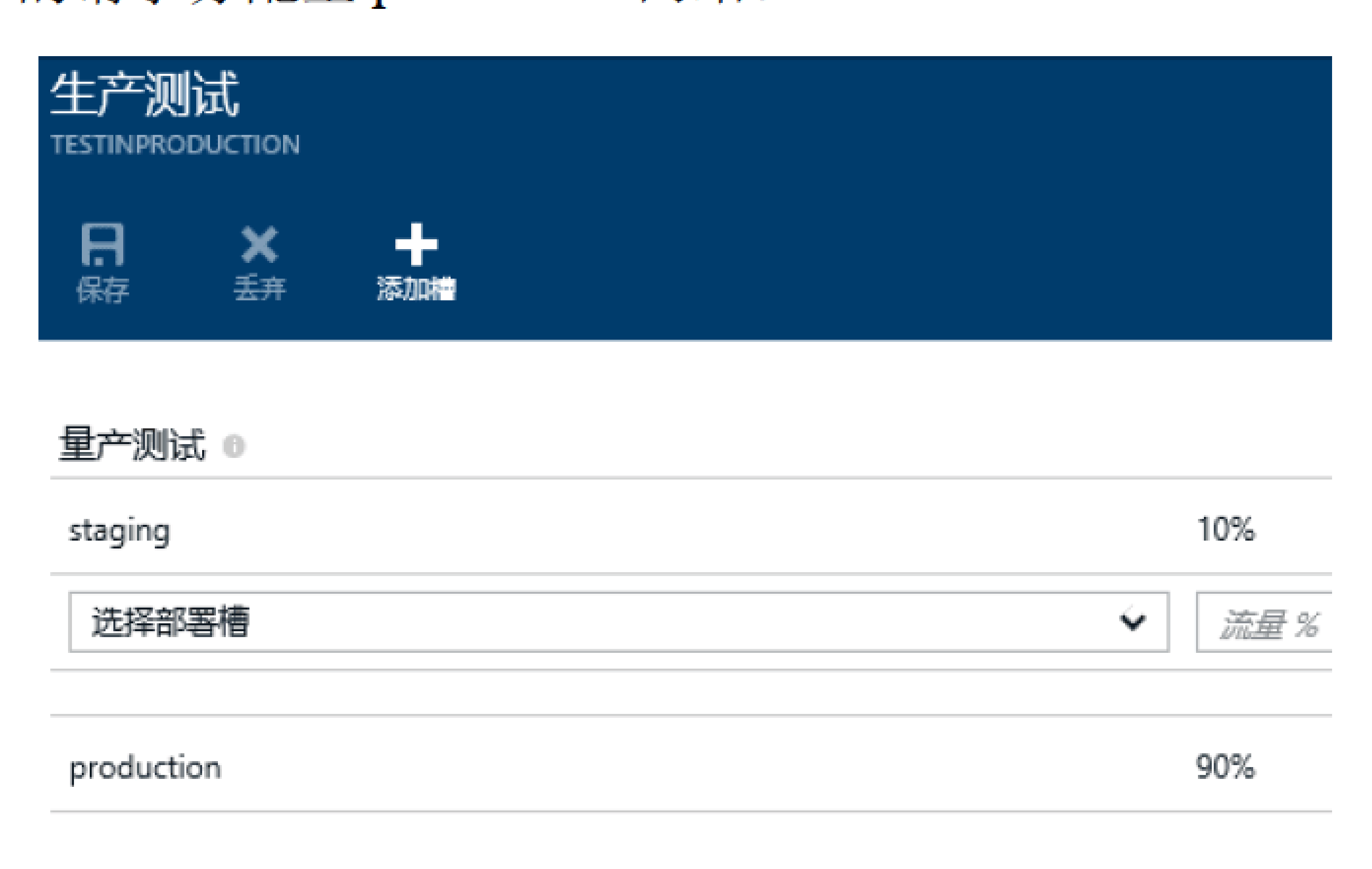


图 5-44　配置流量分配比例

5.8.4　测试

为了验证请求分配情况，需要一些网站请求。有很多工具，比如 Visual Studio 的压力测试完全满足需求。这里推荐一款包含在微软 IIS 6 资源工具包中的轻量级工具 TinyGet。TinyGet 是一个支持多线程和循环的命令行超文本传输协议（HTTP）客户端。下面的命令使用 10 个线程进行测试，每个线程发送 20 个请求。

```
tinyget -srv:testinproduction.azurewebsites.net
    -uri:http://testinproduction.azurewebsites.net
    -x 10 -l 20
```

测试完成后，登录到 Azure 管理门户网站，打开 TestInProduction 的“监视器”页面，如图 5-45 所示，可以看到 TestInProduction 收到了 185 个请求。

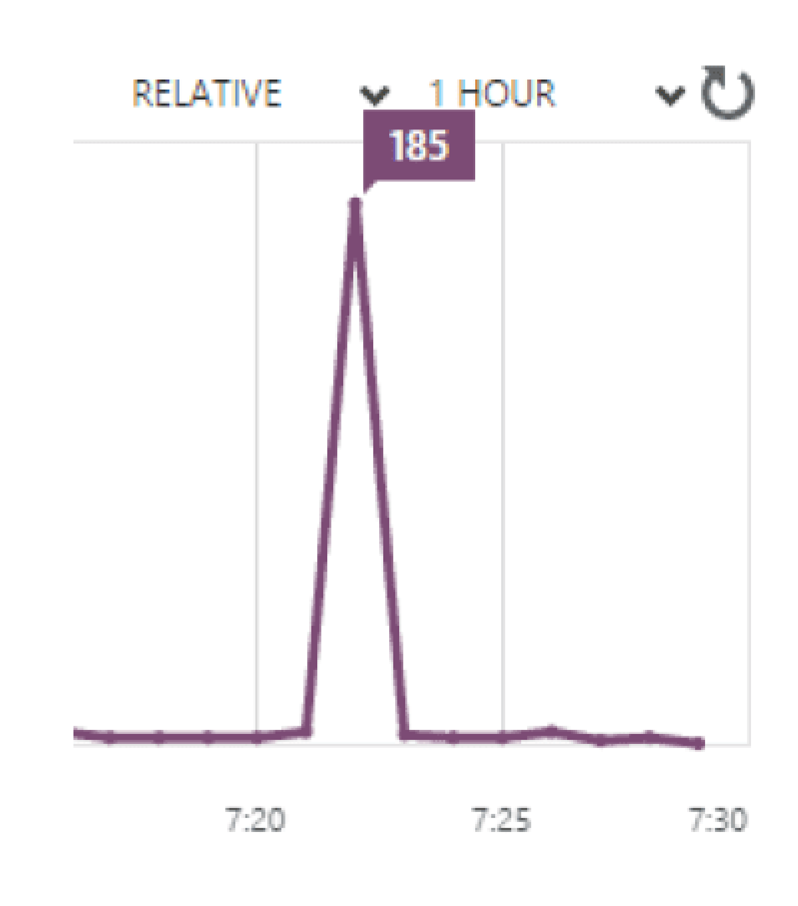


图 5-45　生产网站请求数量

同样，打开 TestInProduction-staging 的“监视器”页面，如图 5-46 所示，可以看到 TestInProduction-staging 收到了 18 个请求。

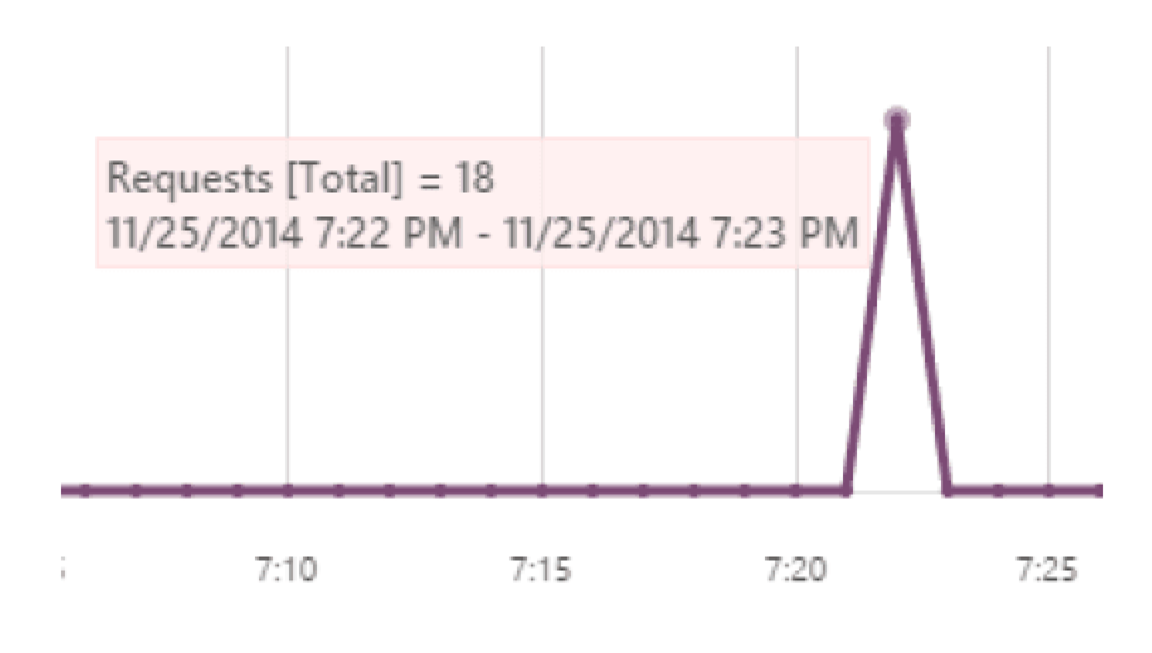


图 5-46　部署槽请求数量

请求分配比例基本是 10%，总数超过 200 个是因为同时打开了浏览器在测试。如图 5-47 所示，可以看到，尽管 URL 指向生产网站 TestInProduction.AzureWebSites.net，但是，请求仍然被转发到了部署槽。

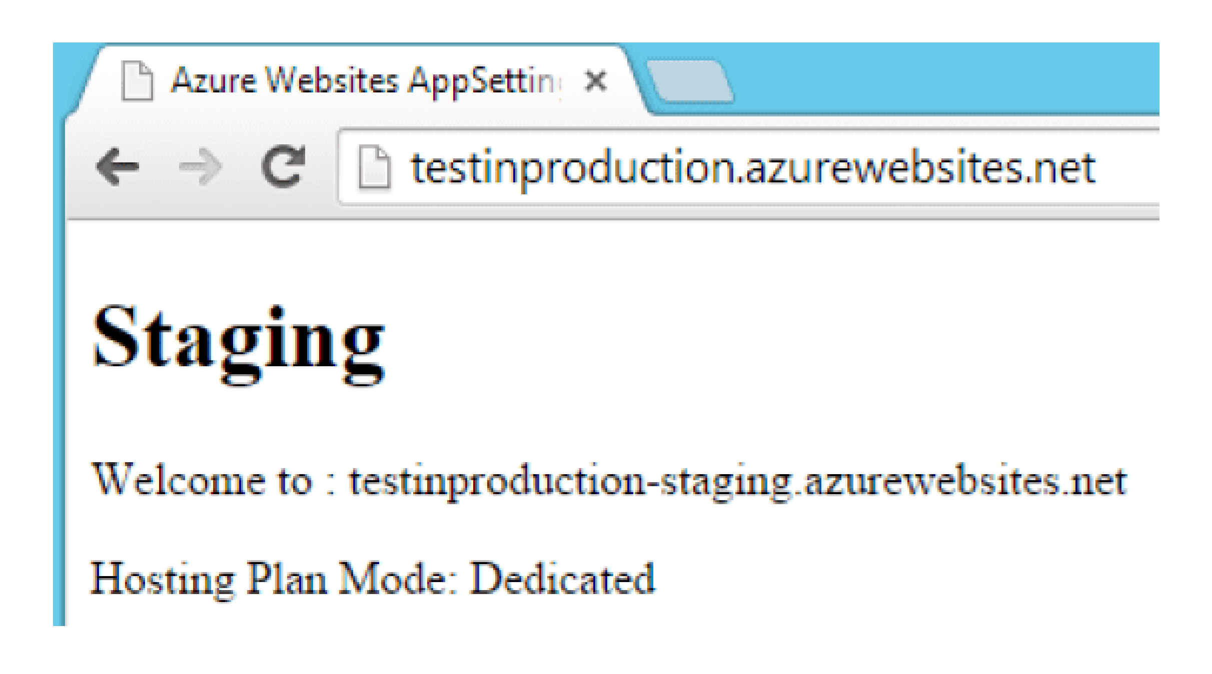


图 5-47　浏览器中查看请求分发结果

5.9　参考文献与扩展阅读

1．Kudu 网站

Kudu 是 Azure 网站 Git 部署、WebJobs 和其他功能的后台引擎。Kudu 是一个开源项目，由微软公司开发和维护。

https://github.com/projectkudu

2．Web Deploy 网站

Web Deploy 可以轻松地将 Web 应用或者 Web 网站部署到 IIS 服务器和 Azure 网站。Visual Studio 和 WebMatrix 集成了 Web Deploy 功能，可以帮助开发人员轻松地将项目部署到 Azure 网站。

http://www.iis.net/downloads/microsoft/web-deploy

3. Web Depoloy 问题排查

该文章介绍了如何排查在使用 Web Deploy 中遇到的各种问题。

http://www.iis.net/learn/publish/troubleshooting-web-deploy

4. Azure 网站部署文档

Azure 网站官方文档，介绍了 Azure 网站支持的部署方式，以及如何选择合适的部署方案将 Web 应用部署到 Azure 网站。

http://azure.microsoft.com/zh-cn/documentation/articles/web-sites-deploy/

5. 从 Git 版本控制系统中部署到 Azure 网站

Azure 网站官方文档，介绍了如何利用 Git 将本地代码库中的 Web 应用部署到 Azure 网站。

http://www.windowsazure.com/en-us/documentation/articles/web-sites-publish-source-control/

6. 持续部署

Azure 网站官方文档，介绍了 Azure 网站支持的持续部署方案，包括 Git、GitHub、TFS 等，以及 Azure 网站如何集成持续部署方案。

http://www.windowsazure.com/en-us/documentation/articles/cloud-services-continous-delivery-use-tfs/

7. WAWSDeploy 工具

David Ebbo 的小工具，基于 Web Deploy，简化了 Web Deploy 命令行部署参数。

http://blog.davidebbo.com/2014/03/WAWSDeploy.html

第 6 章　迁移现有网站到 Azure 网站

如果网站已经运行在企业内部数据中心或者托管在其他的网站服务提供商，都可以轻松地将现有网站迁移到 Microsoft Azure 网站。无论网站采用 PHP+MySQL、ASP.NET+SQL Server、Nodejs、Java 还是不需要数据库的静态网站，都可以将现有网站轻松迁移到 Microsoft Azure 网站，而这一切只需要花费几分钟时间！

在本章，首先讨论网站迁移流程，主要介绍如何在正式迁移网站之前进行兼容性分析。之后，具体讨论如何将 3 种非常典型的网站迁移到 Microsoft Azure 网站。

6.1　网站迁移流程

通常，网站迁移需要遵循下面的流程：

（1）兼容性分析。

（2）创建 Microsoft Azure 网站。

（3）选择合适的网站托管计划：

① 托管计划模式（免费、共享、基本或标准）。

② 实例大小。

③ 实例数目。

（4）根据需要配置网站。

① 基本配置（.NET 版本/PHP 版本等）。

② 选择平台（32 位/64 位）。

（5）迁移 Web 应用。

① 选择合适的迁移技术。

② 迁移文件。

③ 迁移数据库。

（6）功能测试。

（7）配置自有域名，将网站正式切换到 Azure 网站。

6.1.1　兼容性分析

在迁移现有网站之前，要确保 Microsoft Azure 网站能够满足需要。下面是一个兼容性分析列表，在进行迁移之前，必须对照应用需求仔细核对该列表以确保迁移成功。

第 6 章　迁移现有网站到 Azure 网站

如果网站已经运行在企业内部数据中心或者托管在其他的网站服务提供商，都可以轻松地将现有网站迁移到 Microsoft Azure 网站。无论网站采用 PHP+MySQL、ASP.NET+SQL Server、Nodejs、Java 还是不需要数据库的静态网站，都可以将现有网站轻松迁移到 Microsoft Azure 网站，而这一切只需要花费几分钟时间！

在本章，首先讨论网站迁移流程，主要介绍如何在正式迁移网站之前进行兼容性分析。之后，具体讨论如何将 3 种非常典型的网站迁移到 Microsoft Azure 网站。

6.1　网站迁移流程

通常，网站迁移需要遵循下面的流程：

（1）兼容性分析。

（2）创建 Microsoft Azure 网站。

（3）选择合适的网站托管计划：

① 托管计划模式（免费、共享、基本或标准）。

② 实例大小。

③ 实例数目。

（4）根据需要配置网站。

① 基本配置（.NET 版本/PHP 版本等）。

② 选择平台（32 位/64 位）。

（5）迁移 Web 应用。

① 选择合适的迁移技术。

② 迁移文件。

③ 迁移数据库。

（6）功能测试。

（7）配置自有域名，将网站正式切换到 Azure 网站。

6.1.1　兼容性分析

在迁移现有网站之前，要确保 Microsoft Azure 网站能够满足需要。下面是一个兼容性分析列表，在进行迁移之前，必须对照应用需求仔细核对该列表以确保迁移成功。

表 6-1　网站兼容性分析检查列表

<table>
<tr><th>认证方式</th><th>兼容　□</th><th>需要修改代码　□</th></tr>
<tr><td colspan="3">Microsoft Azure 网站支持多种认证方式：
• ASP.NET 表单认证
• 社交网络账号登录（Microsoft Live ID、Google、Yahoo、Facebook 账号）
• Microsoft Azure AD
但是，Microsoft Azure 网站不支持 Windows 集成认证和 Basic 认证。如果使用 Windows 集成认证，可以考虑使用 Azure AD 认证。如果应用使用 Basic 认证，可以考虑用上面任一种认证方式替代。</td></tr>
<tr><th>传输加密</th><th>兼容　□</th><th>需要修改代码　□</th></tr>
<tr><td colspan="3">Microsoft Azure 网站支持 HTTPS。允许上传自己的证书，支持通配符证书（证书名称为 *.mydomain.com）。
目前，Microsoft Azure 网站不支持使用客户端证书的双向 SSL 认证。如果应用要求使用客户端证书，Microsoft Azure 网站暂时不满足要求。</td></tr>
<tr><th>应用框架</th><th>兼容　□</th><th>需要修改代码　□</th></tr>
<tr><td colspan="3">Microsoft Azure 网站支持下面的应用框架：</td></tr>
<tr><th>应用框架</th><th>Azure 支持的版本</th><th>说明</th></tr>
<tr><td>.NET Framework</td><td>2.0（3.0，3.5）
4.0（4.5，4.5.1）</td><td>不支持 1.1 和 1.0</td></tr>
<tr><td>PHP</td><td>5.3，5.4，5.5</td><td>支持客户自定义其他版本的 PHP。具体步骤请参见 4.4 节关于 PHP 的内容</td></tr>
<tr><td>Java</td><td>1.7</td><td>支持 Tomcat 和 Jetty，允许配置自定义容器</td></tr>
<tr><td>Python</td><td>2.7.3</td><td rowspan="2">支持从 0.6 开始的版本，支持自定义版本</td></tr>
<tr><td>Node.JS</td><td>0.10.29</td></tr>
<tr><th>ASP.NET Session 模式</th><th>兼容　□</th><th>需要修改代码　□</th></tr>
<tr><td colspan="3">Microsoft Azure 网站支持多种存储 Session 模式：
• In-Proc（存储在进程内存）
• Azure Cache
• SQL Azure
• 客户自定义
如果应用使用了 Session State Server 模式，迁移到 Microsoft Azure 网站可以采用 Azure Cache 或 SQL Azure 方式。
如果网站计划运行在多个实例上以获取更好的性能和可靠性，不能选用 In-Proc 模式。</td></tr>
<tr><th>安全证书</th><th>兼容　□</th><th>需要修改代码　□</th></tr>
<tr><td colspan="3">有些商业应用要求使用安全证书进行信息加密或者数字签名。标准和基本模式的网站允许从本地的证书库中加载安全证书。
对于免费和共享模式，需要将证书存放在文件系统中。在使用时，从文件系统中加载证书。关于如何从文件系统中加载证书，请参考 MSDN 文档：
http://msdn.microsoft.com/en-us/library/system.security.cryptography.x509certificates.x509certificate2(v=vs.110).aspx
关于安全证书相关的详细内容，请参考 4.3.3 节。</td></tr>
<tr><th>应用包含 COM/COM+模块</th><th>兼容　□</th><th></th></tr>
<tr><td colspan="3">Azure 网站不支持注册客户的 COM/COM+组件。如果应用包含 COM/COM+组件，需要替换这些组件。</td></tr>
</table>

续表

<table>
<tr><th>加载用户配置文件</th><th>兼容　☐</th><th>需要修改代码　☐</th></tr>
<tr><td colspan="3">Microsoft Azure 网站默认不加载用户配置文件(user profile)。如果应用需要加载用户配置文件，那么将其部署到 Microsoft Azure 网站后可能会遇到问题。可以通过指定 Azure 网站应用设置 WEBSITE_LOAD_USER_PROFILE=1 来加载用户配置文件。
详细内容请参考 4.3.3 节。</td></tr>
<tr><th>负载均衡</th><th>兼容　☐</th><th>需要修改代码　☐</th></tr>
<tr><td colspan="3">负载均衡提供了一种有效、透明的方法扩展网络设备和服务器的带宽，增加吞吐量，加强网络数据处理能力，提高网络的灵活性和可用性。如果应用采用了负载均衡，在迁移到 Microsoft Azure 网站时，需要注意会话状态的保持。
Microsoft Azure 网站可以运行多个实例进行负载均衡。负载均衡功能由 Microsoft Azure 网站前端服务器提供。前端服务器采用的负载均衡采用当前请求最少算法将客户请求转发到当前请求数最少的机器。所以，需要考虑将会话状态保存在 Microsoft Azure Cache 或者 SQL Azure 中，而不能采用 In-Proc 方式。</td></tr>
<tr><th>地理位置分布</th><th>兼容　☐</th><th>需要修改代码　☐</th></tr>
<tr><td colspan="3">如果应用要求分布在不同的地理位置，可以采用 Microsoft Azure 流量管理器（Traffic Manager）。Microsoft Azure 流量管理器可以根据客户端的地理位置将客户请求分发到最近的数据中心。流量管理器只支持使用 CNAME 配置自有域名，不支持使用 A 记录。
Microsoft Azure 流量管理器支持 3 种负载平衡方法：
（1）性能：流量被转发到最近的数据中心，以降低网络延迟。
（2）故障切换：根据定义的优先级转发客户请求。所有的请求都被转发到优先级最高的网站。如果优先级最高的网站不可用，则将客户请求转发到优先级次高的网站，以此类推。
（3）轮询：将流量均匀转发到各个数据中心的网站。
请参考 7.3 节了解如何使用 Microsoft Azure 流量管理器。
Microsoft Azure 流量管理器可从以下网址下载：
http://azure.microsoft.com/en-us/services/traffic-manager/</td></tr>
<tr><th>CDN（内容分发网络）</th><th>兼容　☐</th><th>需要修改代码　☐</th></tr>
<tr><td colspan="3">如果应用需要使用内容分发网络提高性能。关于如何在 Microsoft Azure 网站中使用 Azure CDN，请参考 7.5 节。</td></tr>
<tr><th>缓存方案</th><th>兼容　☐</th><th>需要修改代码　☐</th></tr>
<tr><td colspan="3">应用如果采用缓存方案，比如本地缓存或者缓存服务器，迁移到 Microsoft Azure 网站后，可以采用 Microsoft Azure Cache 解决方案。具体信息请参考 7.2 节。</td></tr>
<tr><th>Sandbox（沙漏）</th><th>兼容　☐</th><th>需要修改代码　☐</th></tr>
<tr><td colspan="3">Microsoft Azure 网站是一个多租户环境，同一台虚拟机上可能同时运行不同客户的网站。基于安全考虑，Microsoft Azure 网站引入了额外的安全措施。Sandbox 确保每个网站运行在限定的环境上下文中，与其他用户隔离开。在这种情况下，有些 API 和操作是被禁止的。比如：
• GDI+ API。
• 绑定在本地 TCP 端口。
• 访问其他用户的网站文件。
另外，IPv6 也是受限制的。有些应用（比如 JDBC）默认使用 IPv6。</td></tr>
</table>

6.2　典型应用迁移方案

下面具体讨论如何将以下 3 种非常典型的网站迁移到 Microsoft Azure 网站。

- BlogEngine.NET（ASP.NET 网站）。
- nopCommerce（ASP.NET 网站+SQL Server 数据库）。
- WordPress（PHP 网站+MySQL 数据库）。

6.2.1 BlogEngine.NET（ASP.NET 网站）

BlogEngine.NET 是一个开源 ASP.NET 博客平台。BlogEngine.NET 使用最新的.NET 技术，同时专注于简洁、易用性、可扩展性和创新性的设计。BlogEngine.NET 默认将发布的博客文章保存在文件中，不需要数据库。在很多情况下，这可以显著地节省托管费用。

可以访问下面的网页下载 BlogEngine.NET 应用，或者通过 Web 平台安装程序来下载并安装 BlogEngine.NET。

http://blogengine.codeplex.com/

因为 BlogEngine.NET 不依赖于数据库，所以迁移 BlogEngine.NET 站点极其方便。可以根据实际情况选择 FTP、Git 或者是 Web Deploy。下面具体演示如何将现有的 BlogEngine.NET 站点迁移到 Microsoft Azure 网站。

在开始迁移之前，首先需要登录到 Microsoft Azure 管理门户网站，以“快速创建”方式创建一个新网站。在下面的步骤中以 BlogEngineOnAntares.azurewebsites.net 为例。

6.2.1.1 使用 FTP 迁移网站

FTP 方式简单易用，在迁移不需要后台数据库的网站时非常方便。

（1）如果网站文件不在本地，请先将网站文件下载到本机。

（2）登录到 Microsoft Azure 网站，下载 BlogEngineOnAntares 发布配置文件。

（3）运行 FTP 客户端，比如 FileZilla，连接到 BlogEngineOnAntares 网站的 FTP。FTP 的主机名称和部署凭据（用户名和密码）可以在发布配置文件中找到。

（4）将网站文件上传到/site/wwwroot 目录下即大功告成。

FTP 方式的缺点也是显而易见的，对于 BlogEngine.NET 这种动辄超过 2000 个文件的应用来讲，复制文件大概至少需要 20～30 分钟。

如果对部署时间有要求，可以采用先压缩再上传的方式来节省部署时间。具体操作如下：

（1）在本地将网站文件压缩，如图 6-1 所示，需要在目录下选择所有文件后进行压缩。

（2）运行 FTP 客户端，比如 FileZilla，连接到 BlogEngineOnAntares 网站的 FTP。FTP 的主机名称和部署凭据（用户名和密码）可以在发布配置文件中找到。

（3）将网站文件压缩包上传到/site/wwwroot 目录下。

（4）登录到 SCM 网站，SCM 网站要求认证，需要 Azure 订阅凭据。

SCM 站点的 URL 为 https://SiteName.scm.azurewbsites.net。

（5）登录后，在顶部菜单栏选择 Debug console→CMD 命令。

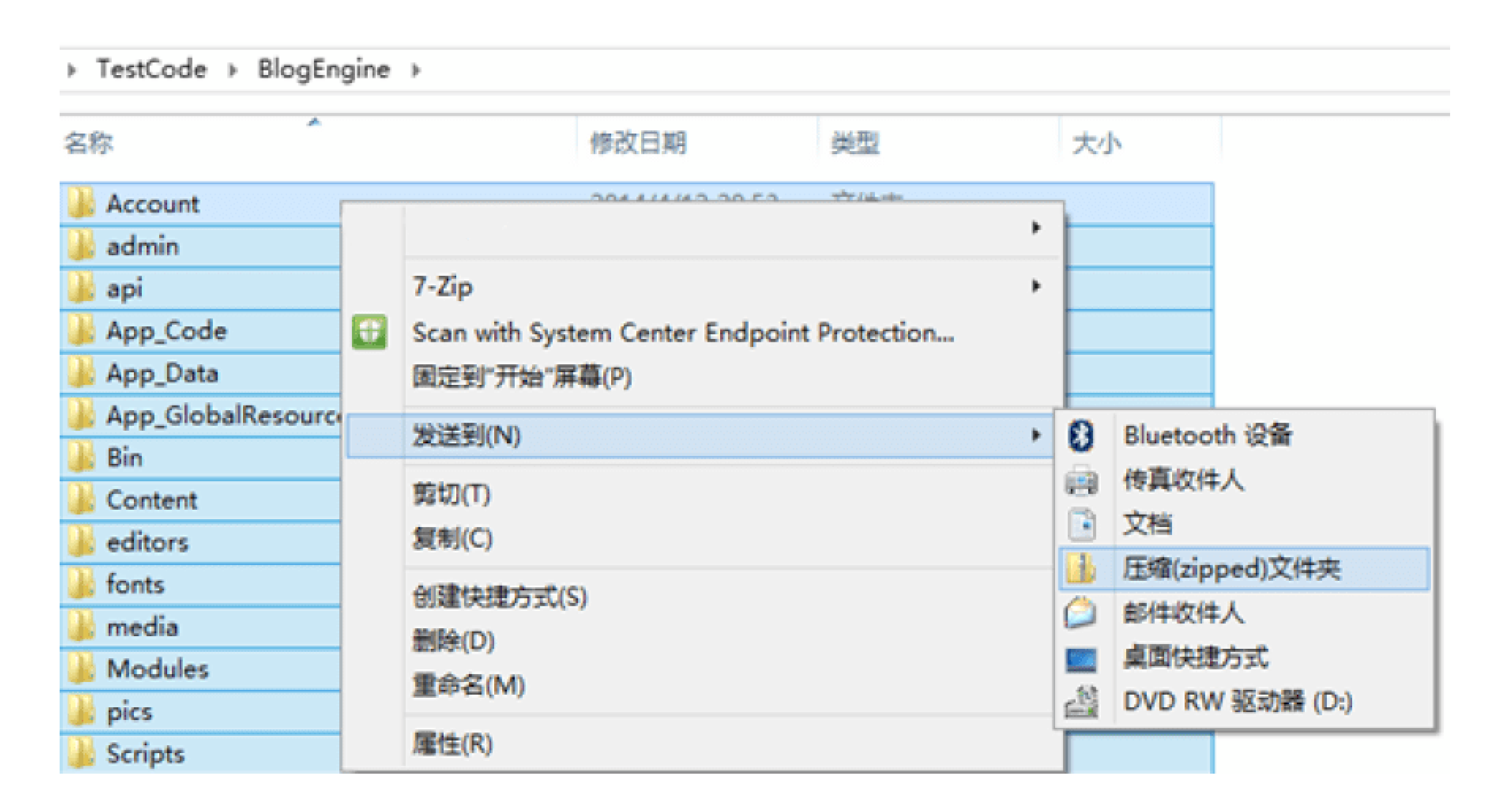


图 6-1　压缩网站文件

（6）如图 6-2 所示，在命令行控制台中进入网站的 wwwroot 目录后运行 unzip 命令解压缩网站文件。

```
Microsoft Windows [Version 6.2.9200]
(c) 2012 Microsoft Corporation. All rights reserved.

D:\home>cd site\wwwroot

D:\home\site\wwwroot>unzip blogengine
```

图 6-2　部署后解压缩网站文件

（7）只需大概十几秒时间即可完成解压缩。

6.2.1.2　使用 Web Deploy 部署

FTP 在上传每个文件时都要首先发送 FTP 命令建立连接，再进行文件传输。相比 FTP，Web Deploy 部署性能要远远好于 FTP。操作步骤如下：

（1）登录到 Microsoft Azure 管理门户网站。

（2）下载目标站点的发布配置文件。

（3）在命令行控制台中将本地文件夹的内容部署到 Microsoft Azure 网站。

运行下面的命令将 c:\BlogEngine.zip 压缩包中的文件同步到 mySite.azurewebsites.net 网站：

```
msdeploy.exe
    -verb:sync
    -source:contentPath="c:\myBlogSite"
    -dest:
        contentPath='mySite',
        ComputerName="https://waws-prod-hk1-001.publish.azurewebsites.
        windows.net:443/msdeploy.axd?site=mySite",
```

```
        UserName='$myBlogOnAzure',
        Password='asdfghjk;lkjhgfd',
        AuthType='Basic'
```

也可以运行 David Ebbo 的 WAWSDeploy 命令行工具：

```
WAWSDeploy.exe C:\msdeploypackage\MyFirstAzureSite c:\publishFile\mysite.
publishSettings
```

6.2.2 nopCommerce（ASP.NET 网站+SQL Server 数据库）

nopCommerce 是非常流行的开源电子商务解决方案。nopCommerce 支持完全定制，具有稳定并且高度可用的特点。NopCommerce 基于 ASP.NET（MVC）技术，后台使用 SQL Server 2005（或更高版本）的数据库。利用 noCommerce 可以轻松地建设一个购物网站，通过互联网销售实物和数字商品。

nopCommerce 的官方网站如下：

http://nopcommerce.codeplex.com/

6.2.2.1 迁移数据库

nopCommerce 后台使用 SQL Server 存储数据。在迁移到 Microsoft Azure 网站时，需要将数据库迁移到 SQL Azure 以获得最好的性能。有很多工具和方法可以协助用户从本地 SQL Server 服务器将数据库迁移到 SQL Azure 数据库。下面具体讨论两种方法：使用 SQL Server 管理工具和 Web Deploy。如果对其他方法感兴趣，请参考下面的文档：

Migrating Databases to Microsoft Azure SQL Database

http://msdn.microsoft.com/en-us/library/ee730904.aspx

1. 使用 SQL Server 数据库管理工具迁移数据库

（1）创建一个 SQL Azure 数据库服务器。如果想使用已有的 SQL Azure 数据库服务器，请忽略此步骤。第 2 章中介绍过，可以通过自定义创建网站的方式同时创建网站和 SQL Azure 数据库。下面的步骤介绍如何单独创建一个 SQL Azure 数据库。

① 登录到 Microsoft Azure 管理门户网站。

② 在左侧导航栏选择 SQL 数据库。

③ 在右侧的 sql 数据库页面单击顶部的“服务器”。

④ 在底部命令栏单击“添加”。

⑤ 如图 6-3 所示，提供创建数据库服务器需要的信息，包括以下几项：

- 区域：指定 SQL 数据库服务器所在的数据中心。
- 输入登录名和密码。
- 选择“允许 WINDOWS AZURE 服务访问服务器”复选框。

⑥ 单击“完成”按钮，创建数据库。

（2）修改防火墙规则，允许当前 IP 地址访问新创建的数据库服务器。

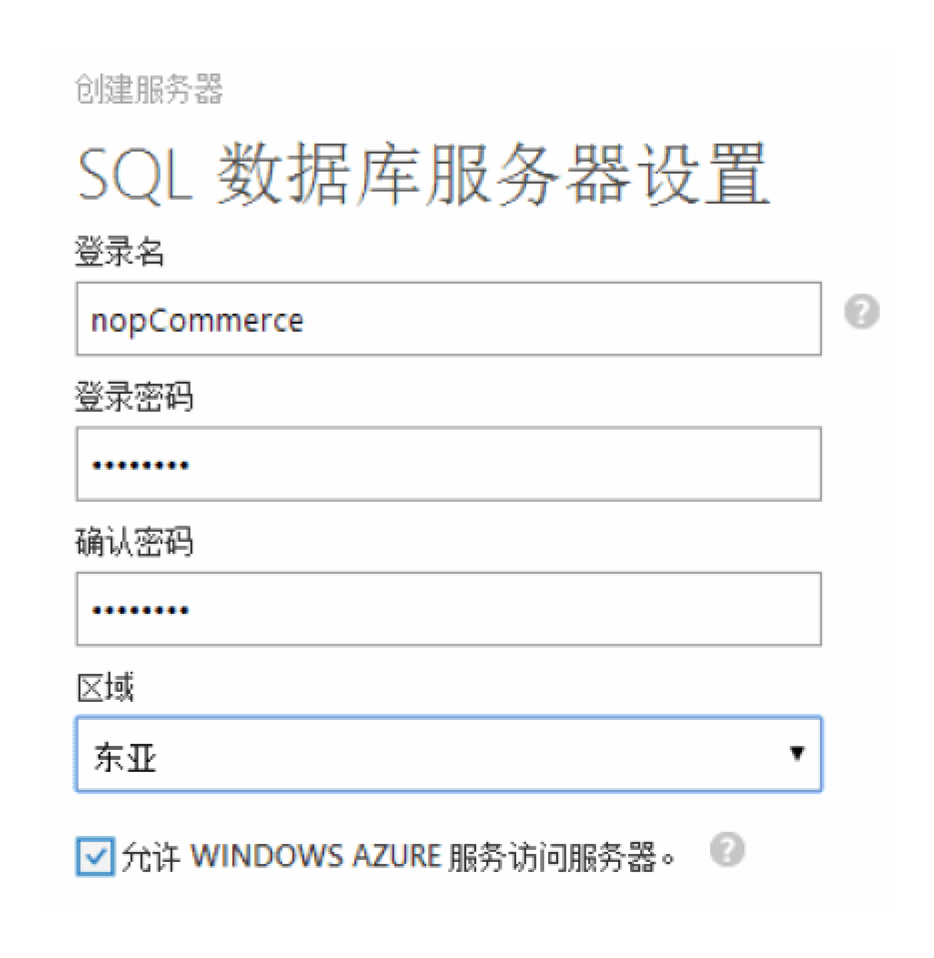


图 6-3　创建 SQL Azure 数据库

① 数据库服务器创建成功后，在 Azure 管理门户网站选择新创建的数据库服务器。

② 单击顶部的“配置”，如图 6-4 所示，将当前客户端 IP 地址添加到允许的 IP 地址。



图 6-4　指定允许访问 SQL Azure 的 IP 地址

（3）使用 SQL Server 数据库管理器迁移数据库到 SQL Azure 数据库。

① 运行 SQL Server 数据库管理器，连接到本地数据库。

② 如图 6-5 所示，右击本地数据库，选择“任务”→“将数据库部署到 SQL Azure”命令。



图 6-5　将数据库部署到 SQL Azure

③ 如图 6-6 所示，在“部署设置”窗口，单击“连接”按钮，打开“连接到服务器”对话框，输入服务器名称和凭据后，单击“连接”按钮。



图 6-6　连接到 SQL Azure 数据库

④ 如图 6-7 所示，在“部署设置”窗口，指定新数据库名称和数据库版本等信息。

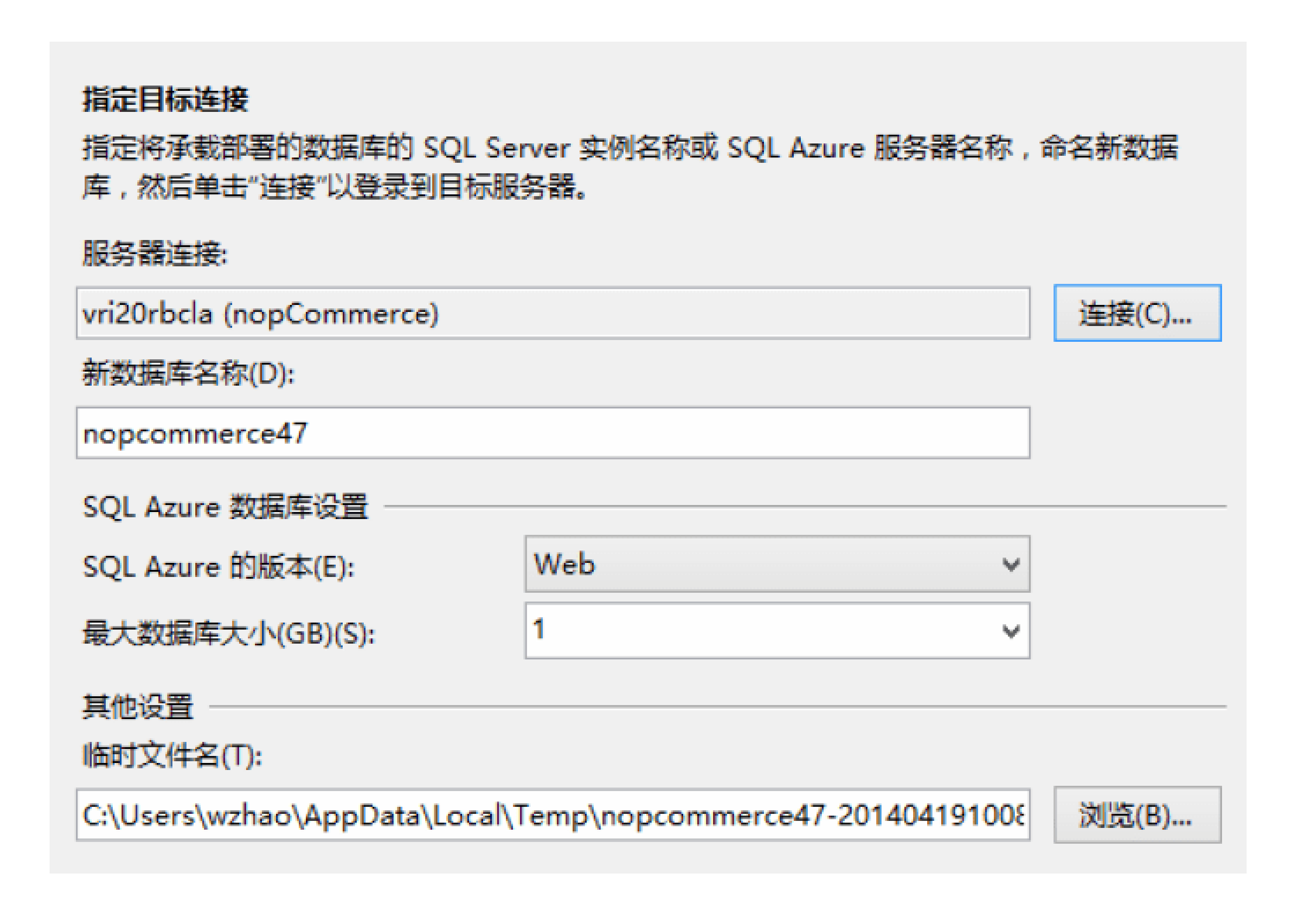


图 6-7　指定目标数据库

⑤ 单击“下一步”按钮，确认源数据库和目标数据库信息，单击“完成”按钮开始迁移数据库。迁移过程如下：

- 将本地数据库导出到本地临时数据包。
- 在 SQL Azure 数据库服务器上创建指定的目标数据库。
- 将本地临时数据包导入到目标服务器。

2. 使用 Web Deploy 迁移数据库

Web Deploy 提供了 dbFullSql Provider，能够将本地 SQL Server 数据库/开发数据库部署到一个远程托管 SQL Server 数据库。该功能在第一次部署或者迁移网站和数据库时非常实用。

dbFullSql Provider 使用 SQL Server 管理对象（SMO）将源数据库工程导出为基于 Transact-SQL DDL（数据定义语言）和 DML（数据操作语言）的脚本。然后在目标数据库上执行这些脚本还原数据。下面的命令将本地运行的 nopCommerce 数据库复制到 SQL Azure 服务器。如果 SQL Azure 服务器上没有 nopCommerce 数据库，那么部署过程中会自动创建 nopCommerce 数据库。该命令只需指定源数据库和目标数据库的连接字符串。由于 SQL Azure 的特殊性，需要特别指定目标数据库的版本（TargetServerVersion）和引擎类型（TargetDatabaseEngineType）。

```
msdeploy.exe -verb:sync
-source:dbFullSql="Data Source=localhost;Integrated Security=true;
        Initial Catalog=nopCommerce",
        TargetServerVersion=Version110,
        TargetDatabaseEngineType=SqlAzureDatabase
-dest:dbFullSql="Server=tcp:vri20rbcla.database.windows.net,1433;
       Database=nopcommerce;
       User ID=nopCommerce@vri20rbcla;Password=P@ssw0rd!;
       Trusted_Connection=False;Encrypt=True;Connection Timeout=30;"
```

6.2.2.2　迁移网站

如前面迁移 BlogEngine.NET 相同，可以通过 FTP、Web Deploy/WAWSDeploy 命令行或者 Git 将 nopCommerce 网站内容同步到 Microsoft Azure 网站。

6.2.2.3　注意事项

nopCommerce 的数据库连接字符串没有保存在 web.config 中，而是保存在 /App_Data/settings.txt 文件中。因此使用 Web Deploy 部署时也无法检测到数据库连接字符串，在部署过程中也无法进行修改。可以通过下列方法修改：

（1）手工修改/App_Data/settings.txt 文件中的数据库连接字符串，将其指向 SQL Azure 数据库。如果网站只需部署一次，那么可以选择手工修改。如果在本地部署一份开发、测试环境，指向本地数据库，在 Microsoft Azure 网站上运行一个网站指向 SQL Azure 数据库，需要经常部署网站更新，那么需要考虑下面的自动化步骤，以简化部署过程并避免出错。

（2）使用 Web Deploy 的 preSync 或者 postSync 功能自动修改。preSync 指定在部署前运行的命令或者脚本，postSync 指定部署结束后运行的命令或者脚本。因此，可以写一个脚本自动在部署之前替换数据库连接字符串。具体信息可以参考下面的文档：

http://technet.microsoft.com/en-us/library/ee619740(v=ws.10).aspx

Web Deploy 支持部署过程中的查找与替换，具体请参考下面文档中介绍的 replace 功能：

http://technet.microsoft.com/en-us/library/dd569089(v=ws.10).aspx

6.2.2.4　修改相关的应用配置

迁移完成后，可能需要修改相关的 nopCommerce 配置，比如网络商店的 URL 等，可以用 admin 用户登录 nopCommerce 网站，访问下面的 URL 修改商店信息：

http://yoursite/Admin/Store/List

其他相关配置请参考 nopCommerce 的文档。

6.2.3　WordPress（PHP 网站+MySQL 数据库）

WordPress 是一个强大的开源内容发布平台。WordPress 基于 PHP 和 MySQL。使用 WordPress 可以搭建功能强大的网络信息发布平台，WordPress 能让用户集中精力做好网站的内容而无须关心网站的后台技术。

WordPress 的官方网站如下：

http://wordpress.org/

WordPress 后台使用 MySQL 数据库。目前，WordPress 也支持后台运行 SQL Server，在 Azure 网站的 Web 应用库中，可以安装 Brandoo WordPress。该版本支持 SQL 和 SQL Azure 数据库。也可以参考下面的网页：

http://www.microsoft.com/web/wordpress/

前面已经讨论了如何将 SQL Server 迁移到 SQL Azure，下面讨论如何将 WordPress+MySQL 网站迁移到 Microsoft Azure 网站。

6.2.3.1　创建 MySQL 数据库

可以通过 Microsoft Azure 网站的自定义新建网站方式同时新建一个网站和一个 MySQL 数据库或者 SQL Server 数据库，并将其与创建的网站关联。下面的步骤创建一个网站，并创建一个 MySQL 数据库。

（1）登录到 Microsoft Azure 管理门户网站。

（2）单击左下角的“新建”按钮，选择“计算”→“网站”。

（3）选择“自定义创建”，如图 6-8 所示，提供如下信息：

图 6-8　创建网站

① URL：输入要创建的网站名称。该名称必须是未被注册的名称。

其他相关配置请参考 nopCommerce 的文档。

6.2.3 WordPress（PHP 网站+MySQL 数据库）

WordPress 是一个强大的开源内容发布平台。WordPress 基于 PHP 和 MySQL。使用 WordPress 可以搭建功能强大的网络信息发布平台，WordPress 能让用户集中精力做好网站的内容而无须关心网站的后台技术。

WordPress 的官方网站如下：

http://wordpress.org/

WordPress 后台使用 MySQL 数据库。目前，WordPress 也支持后台运行 SQL Server，在 Azure 网站的 Web 应用库中，可以安装 Brandoo WordPress。该版本支持 SQL 和 SQL Azure 数据库。也可以参考下面的网页：

http://www.microsoft.com/web/wordpress/

前面已经讨论了如何将 SQL Server 迁移到 SQL Azure，下面讨论如何将 WordPress+MySQL 网站迁移到 Microsoft Azure 网站。

6.2.3.1 创建 MySQL 数据库

可以通过 Microsoft Azure 网站的自定义新建网站方式同时新建一个网站和一个 MySQL 数据库或者 SQL Server 数据库，并将其与创建的网站关联。下面的步骤创建一个网站，并创建一个 MySQL 数据库。

（1）登录到 Microsoft Azure 管理门户网站。

（2）单击左下角的“新建”按钮，选择“计算”→“网站”。

（3）选择“自定义创建”，如图 6-8 所示，提供如下信息：

新网站 - 自定义创建
创建网站
URL
wzhaoCustomCreate .azurewebsites.net
WEB 宿主计划
创建新的 Web 宿主计划
区域
东亚
数据库
创建免费的 20 MB SQL 数据库
数据库连接字符串名称
DefaultConnection
从源代码管理发布

图 6-8 创建网站

① URL：输入要创建的网站名称。该名称必须是未被注册的名称。

② WEB 宿主计划：可以选择已有的 Web 宿主计划，或者选择“创建新的 Web 宿主计划”。

③ 区域：选择一个数据中心，原则是尽量靠近网站的客户。

④ 数据库：有以下选项：

- 使用现有的 SQL 数据库。
- 创建新的 SQL 数据库。
- 使用现有的 MySQL 数据库。
- 创建新的 MySQL 数据库。

在这里选择创建新的 MySQL 数据库。

⑤ 指定数据库连接字符串名称。

（4）如图 6-9 所示，指定 MySQL 数据库的名称和区域，单击“完成”按钮。



图 6-9　新建 MySQL 数据库

还可以单独创建一个 MySQL 数据库，步骤如下：

（1）登录到 Microsoft Azure 管理门户网站。

（2）单击左下角“新建”按钮。

（3）选择“商店”。

（4）如图 6-10 所示，在外接程序列表中选择 ClearDB MySQL Database，单击“下一步”按钮。

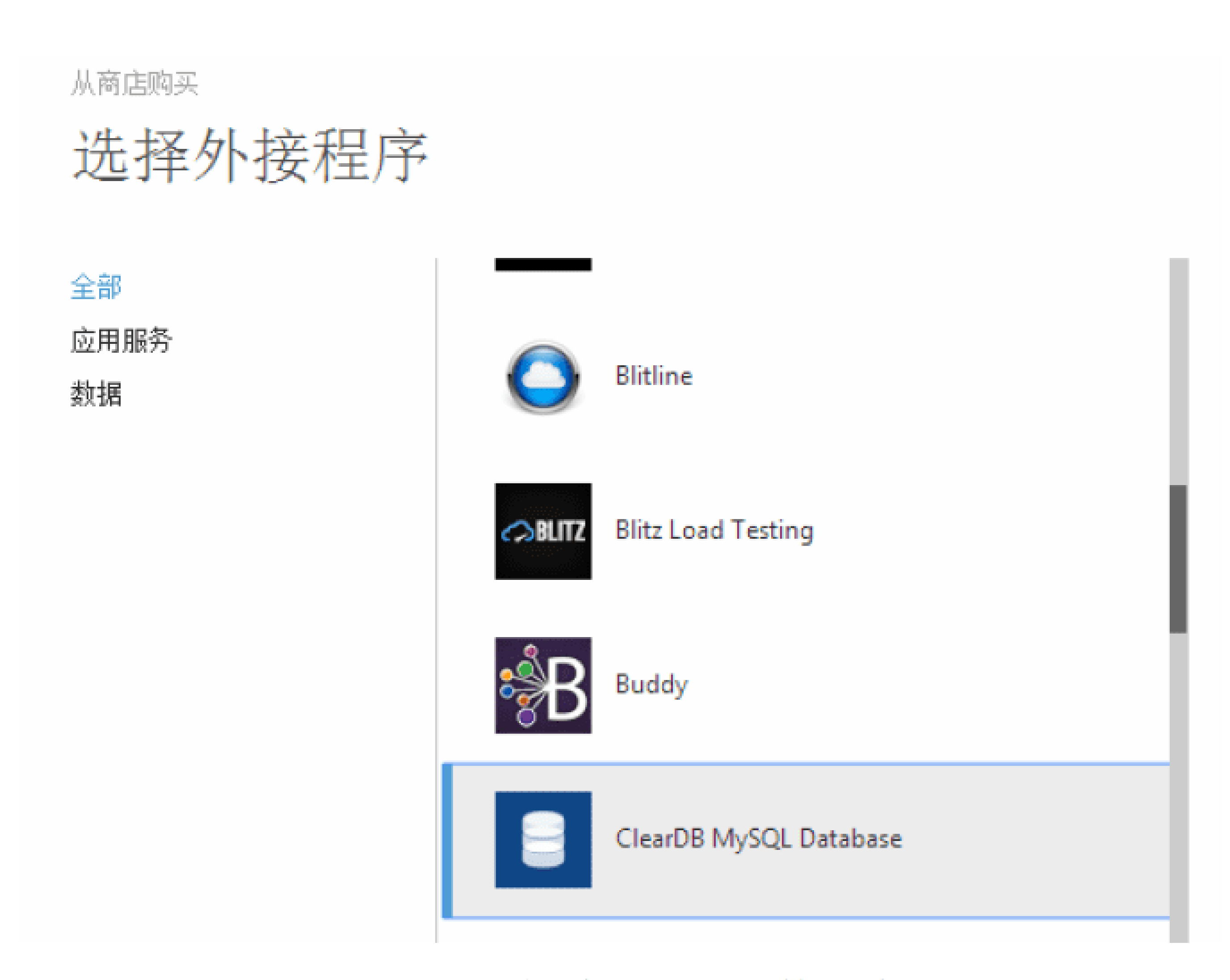


图 6-10　新建 MySQL 数据库

（5）如图 6-11 所示，在“个性化外接程序”中，根据应用的需要选择合适的计划，并指定数据库名称。单击“下一步”按钮，完成数据库的创建。

图 6-11　选择 MySQL 计划

6.2.3.2　迁移 MySQL 数据库

1．使用 MySQL 命令行工具

MySQL 提供了几个实用的命令行工具。其中，mysqldump 命令行工具可用于转储数据库或用于备份或转移到另一个 SQL 服务器（不一定是一个 MySQL 服务器），与 mysql 命令行工具或者 mysqlimport 命令行工具结合，可以方便地将 MySQL 数据库从本地迁移到 Microsoft Azure 中的 MySQL 数据库。

下面的命令将本地的 skywp_db 数据库完整复制到 Microsoft Azure 中的 MySQL 数据库。其中，mysqldump 命令用于将数据库完整导出，并作为 mysql 命令的输入。mysql 命令将根据导出的数据库信息在 Microsoft Azure 的 MySQL 数据库上完整重建，新数据库的名称为 skywponazure_db 指定的数据库服务器上。

mysqldump.exe" --opt -h <LocalServerName> -u<*LocalDBUser*> -p<*LocalDBPassword*> skywp_db mysql -h ap-cdbr-azure-east-a.cloudapp.net –u<RemoteDBUser>-p <RemoteDBPassword>-C wzhaowordpress

其中，mysqldump 命令中，-h 用于指定源数据库服务器；mysql 命令中，-h 指定目标数据库服务器；-C 指定在传输过程中进行压缩以提高传输速度。可以在发布配置文件中获取目标数据库的服务器名称和用户凭据，也可以通过连接门户网站查看 MySQL 数据库连接字符串源获取相关信息。

如果通过自定义创建网站，并同时创建了 MySQL 数据库，可以在配置页面找到 MySQL

数据库的服务器名称、用户名和密码。

- 登录到 Microsoft Azure 管理门户网站。
- 打开创建的网站。
- 单击顶部的“配置”，打开“配置”页面。
- 如图 6-12 所示，在“连接字符串”区域，单击“显示连接字符串”，可以查看新建的 MySQL 数据库的信息。

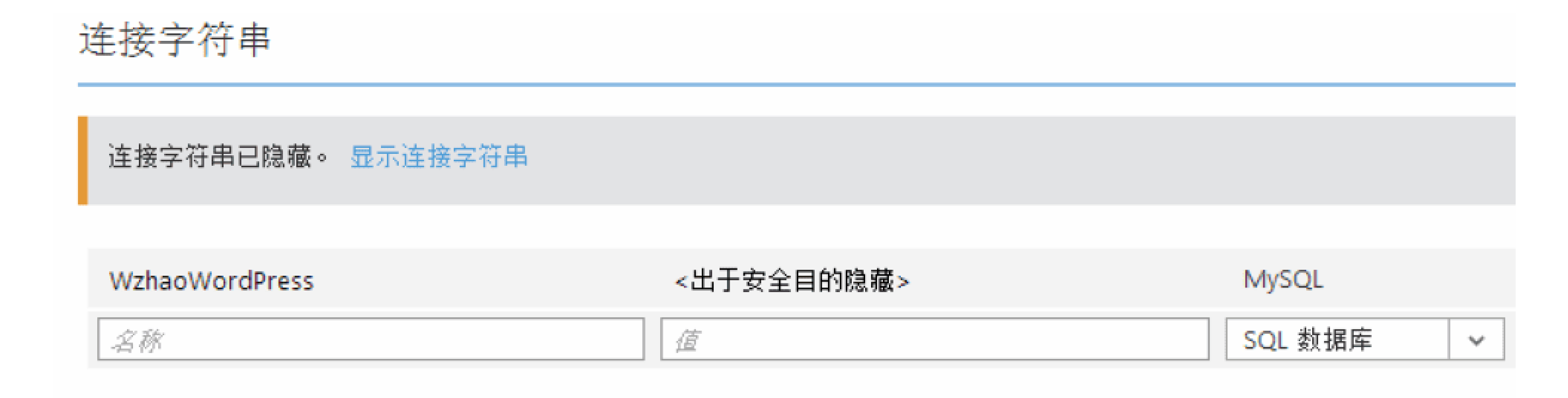


图 6-12　连接字符串

2. 使用 Web Deploy 命令行

使用 Web Dpeloy 的 dbFullMySql 扩展模块，可以将本地运行的 MySQL 数据库迁移到运行在 Microsoft Azure 中的 MySQL 数据库。具体信息请参考以下网页：

http://blogs.iis.net/msdeploy/archive/2009/03/30/msdeploy-sample-custom-provider-to-sync-mysql-databases-dbfullmysql.aspx

3. 使用 MySQL Workbench

MySQL Wrokbench 是管理 MySQL 的 UI 工具。它提供了数据导出和导入功能，这些功能基于 mysqldump.exe 和工具。通过 MySQL Workbench 的数据导出和导入功能，可以轻松也将 MySQL 数据库迁移到 Microsoft Azure 上。

关于 MySQL Workbench 的数据导出和导入功能，请参考 MySQL 的文档：

http://dev.mysql.com/doc/workbench/en/wb-mysql-connections-navigator-management-data-export.html

http://dev.mysql.com/doc/workbench/en/wb-mysql-connections-navigator-management-data-import-restore.html

6.2.3.3　迁移网站内容文件

如前面所讲，可以通过 FTP、Web Deploy 或者 Git 将网站文件部署到 Microsoft Azure 网站。

6.2.3.4　修改 WordPress 应用配置

首先，需要修改数据库连接字符串。WordPress 的数据库连接信息保存在/wp-config.php 中，需要修改下面的信息：

```
define('DB_NAME', '新的 MySQL 数据库名称');
define('DB_USER', '连接到新 MySQL 数据库的用户名称');
define('DB_PASSWORD', 'password');
define('DB_HOST', '新的 MySQL 数据库服务器');
```

另外，如果 Wordpress 使用了新的域名，如图 6-13 所示，需要修改 WordPress 的站点配置。

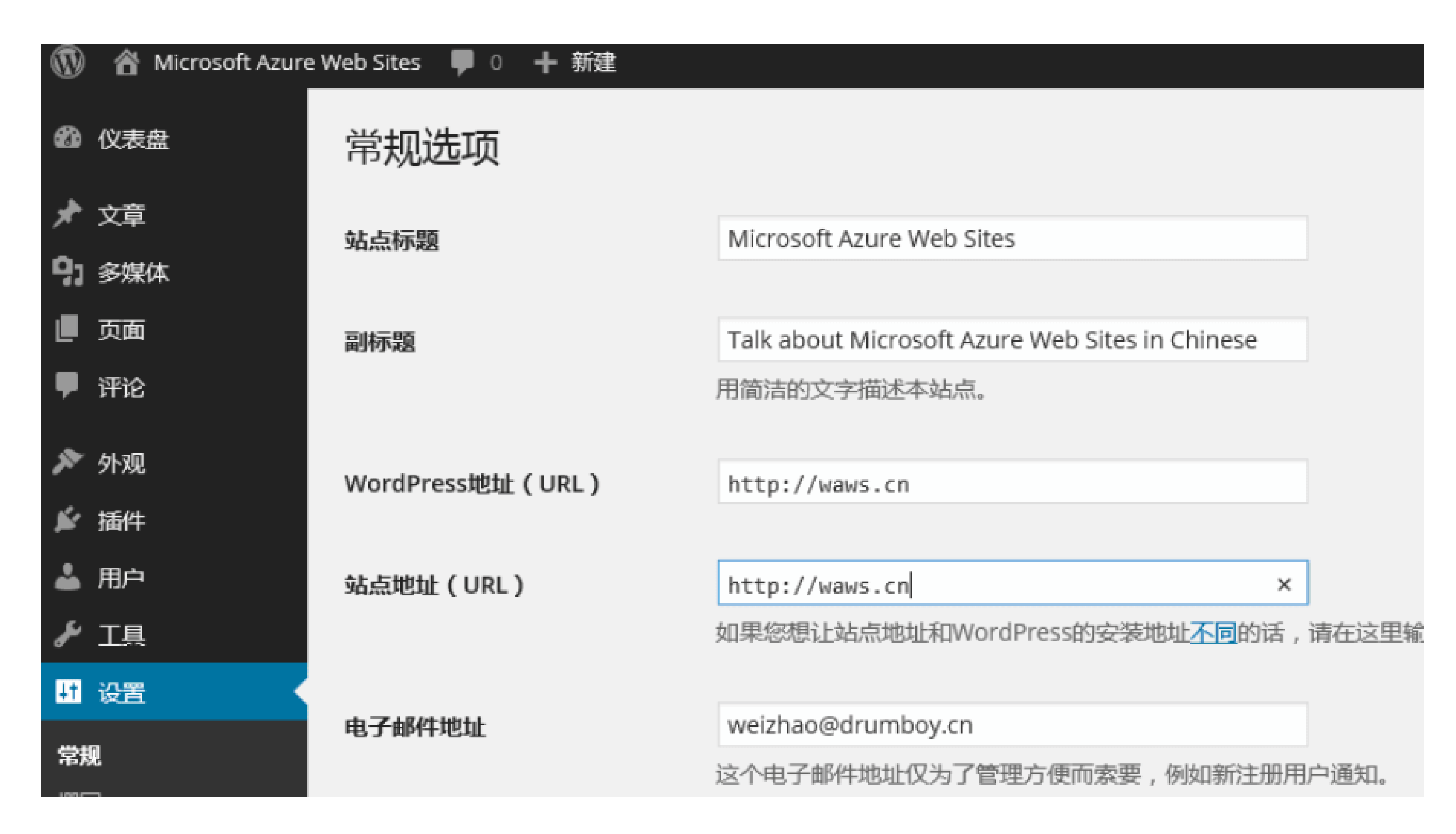


图 6-13　WordPress 设置

其他需要修改的 WordPress 选项请参考 WordPress 文档。

6.3　网站迁移工具

为了帮助客户从本地环境将网站迁移到 Azure 网站，Azure 网站产品组开发了一个迁移工具。该工具可以协助客户将运行在本地的网站以及数据库轻松迁移到 Azure 网站。目前该工具仅支持迁移运行在本地的 IIS 服务器上的 Web 应用，其他环境暂时不支持。迁移过程主要包括 3 个步骤：

（1）安装迁移工具。

（2）运行兼容性分析。

（3）迁移网站。

在本节，将详细介绍如何使用该工具。

6.3.1　安装 Azure 网站迁移工具

访问 http://www.MoveMeToTheCloud.net 网站，如图 6-14 所示，单击 Dedicated IIS Server 按钮后，会导向安装页面，遵循安装向导，安装 Azure Websites Migration Assistant 工具。

Migrate to Azure in 3 easy steps

1. Install the Migration Assistant tool

Select the option that describes your website hosting. (More options coming soon!)

Dedicated IIS Server

2. Run the readiness assessment

3. Migrate your site(s)

图 6-14　安装迁移工具

6.3.2　兼容性分析

安装完成后，该工具自动运行。可以选择连接到本地计算机或者一台远程 IIS 服务器。该工具要求管理员权限，连接到远程服务器，需要提供计算机名称以及管理员用户名和密码。

如图 6-15 所示，该工具自动列出指定的 IIS 服务器上运行的所有网站。在本例中，将运行在本地的一个 EISK 应用迁移到 Azure 网站。该应用是一个 ASP.NET 的示例程序，包括一个 ASP.NET Web 应用和后台数据库。该应用的源代码可以从如下网站获取：

http://eisk.codeplex.com/

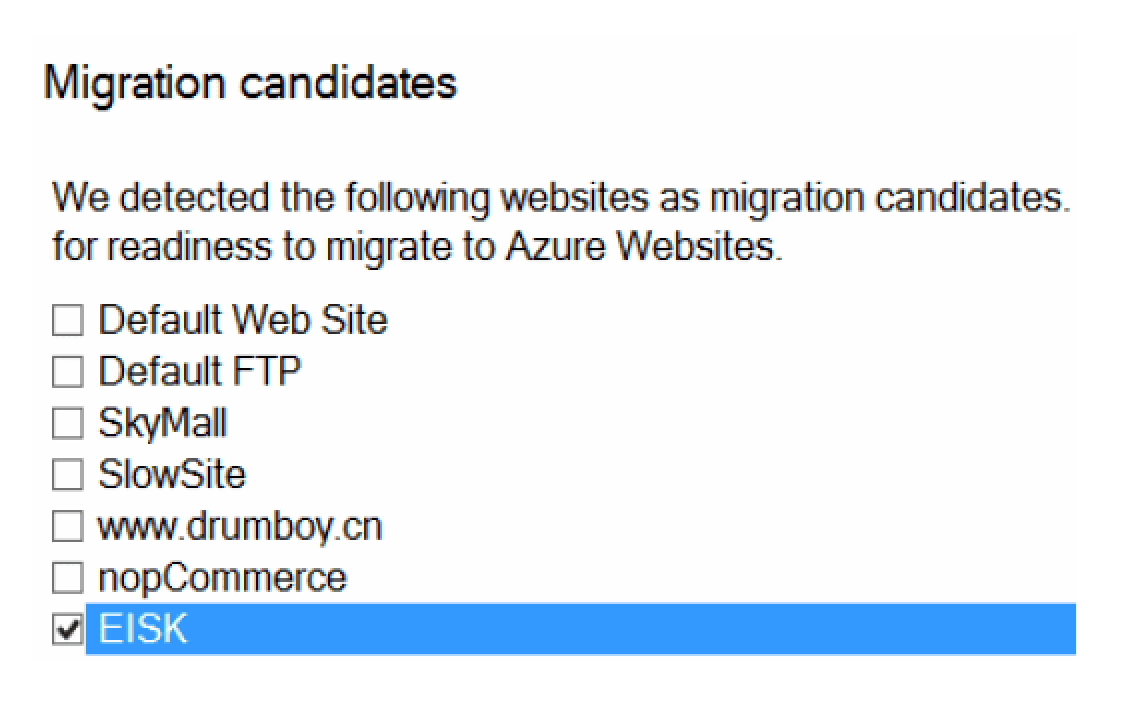


图 6-15　选择要迁移的网站

选择 EISK 网站后，单击 Next 按钮。该工具分析网站的配置和应用代码的兼容性，并生成兼容性报告。兼容性分析包含如下项目：

（1）IIS 绑定。Azure 网站支持 80 端口（HTTP）和 443 端口（HTTPS）。如果本地网站使用了其他端口，在移植后，只能使用 80 端口。

（2）认证方法。Azure 网站支持匿名认证和表单认证，同时支持通过 Azure 活动目录实现 SSO（Single-Sign On）。

（3）使用安装在 GAC（Global Assembly Cache，全局程序集缓存）里的应用程序集。

在前面介绍过，Azure 网站的运行环境只包含.NET 框架。如果应用引用了其他的 Assembly，需要将该 Assembly 部署到 Azure 网站的 bin 目录下。

（4）IIS 5 兼容模式。IIS 6（Windows 2003 服务器）支持 IIS 5 兼容模式。该模式允许网站应用运行在 IIS 5 模式下。Azure 网站不支持 IIS 5 兼容模式。

（5）IIS 配置。在 Azure 网站中，有些 IIS 配置项不允许修改。如果需要修改这些配置

项，需要使用 8.3 节介绍的应用配置转换技术。

（6）每个网站对应一个应用程序池。

Azure 网站中，每个网站对应一个应用程序池。同一网站下的所有应用和虚拟目录等都使用相同的应用程序池。如果要求不同的应用和虚拟目录采用不同的应用程序池，可以考虑将这些应用部署在不同的网站。

（7）COM/COM+组件。Azure 网站不允许注册 COM/COM+组件，如果应用使用了 COM/COM+组件，需要将这部分代码替换掉。

（8）ISAPI Filters。Azure 网站支持使用 ISAPI Filter，但是需要将对应的 DLL 文件部署到网站 bin 目录下，并通过 Web.config 进行注册。

（9）其他组件。比如 SharePoint、FrontPage Server Extensions (FPSE)、FTP、SSL 证书等将不会被迁移。

如图 6-16 所示，迁移工具自动将对应的需要注意的兼容性问题列出。

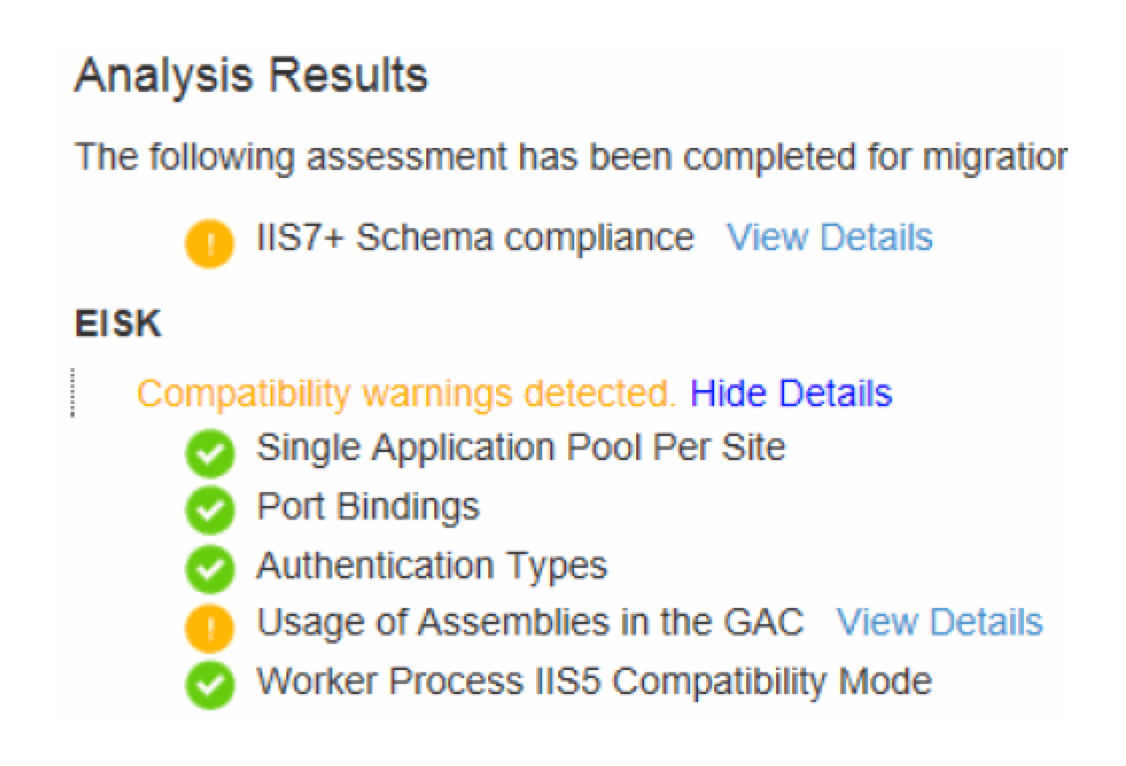


图 6-16　兼容性分析报告

6.3.3　迁移网站

兼容性分析完成后，可以开始迁移网站。如图 6-17 所示，首先使用 Azure 账户登录。登录后，如果包含多个订阅，需要选择要使用的 Azure 订阅。同时需要指定 Azure 数据中心。Azure 网站和数据库将被创建在指定的数据中心。

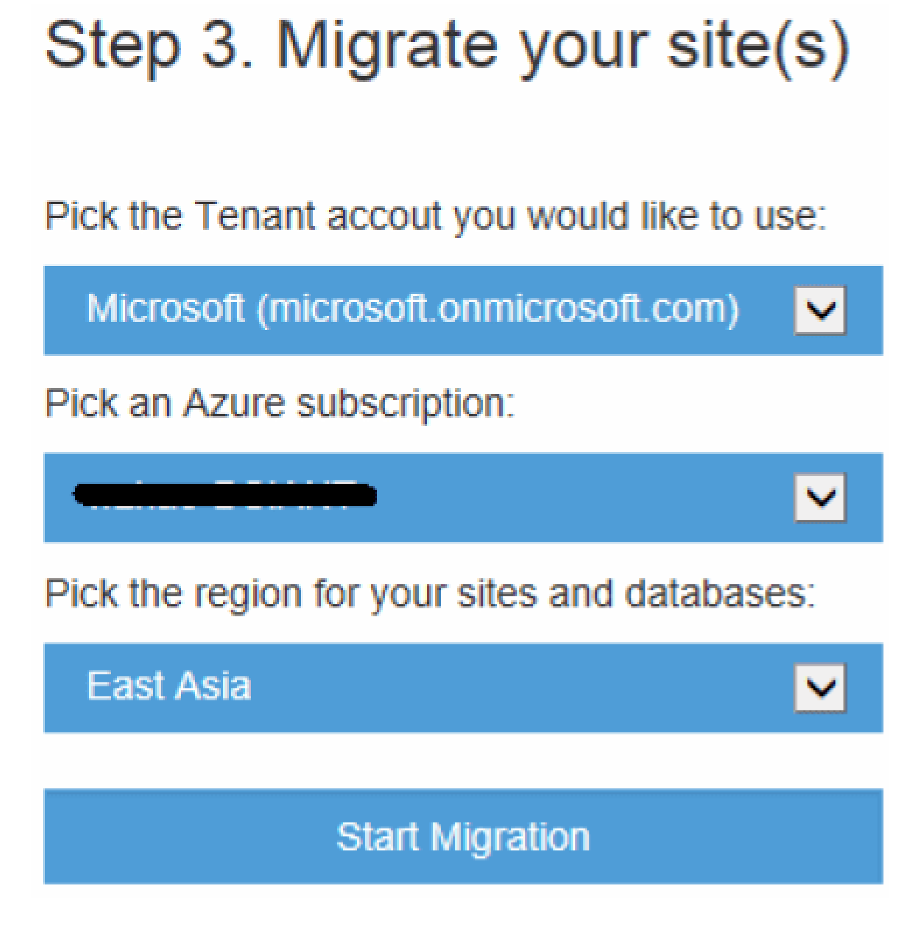


图 6-17　选择 Azure 订阅和数据中心

单击 Start Migration，如图 6-18 所示指定网站的名称和 SQL 服务器的相关信息。SQL 服务器可以选择新建或者使用已有的服务器。

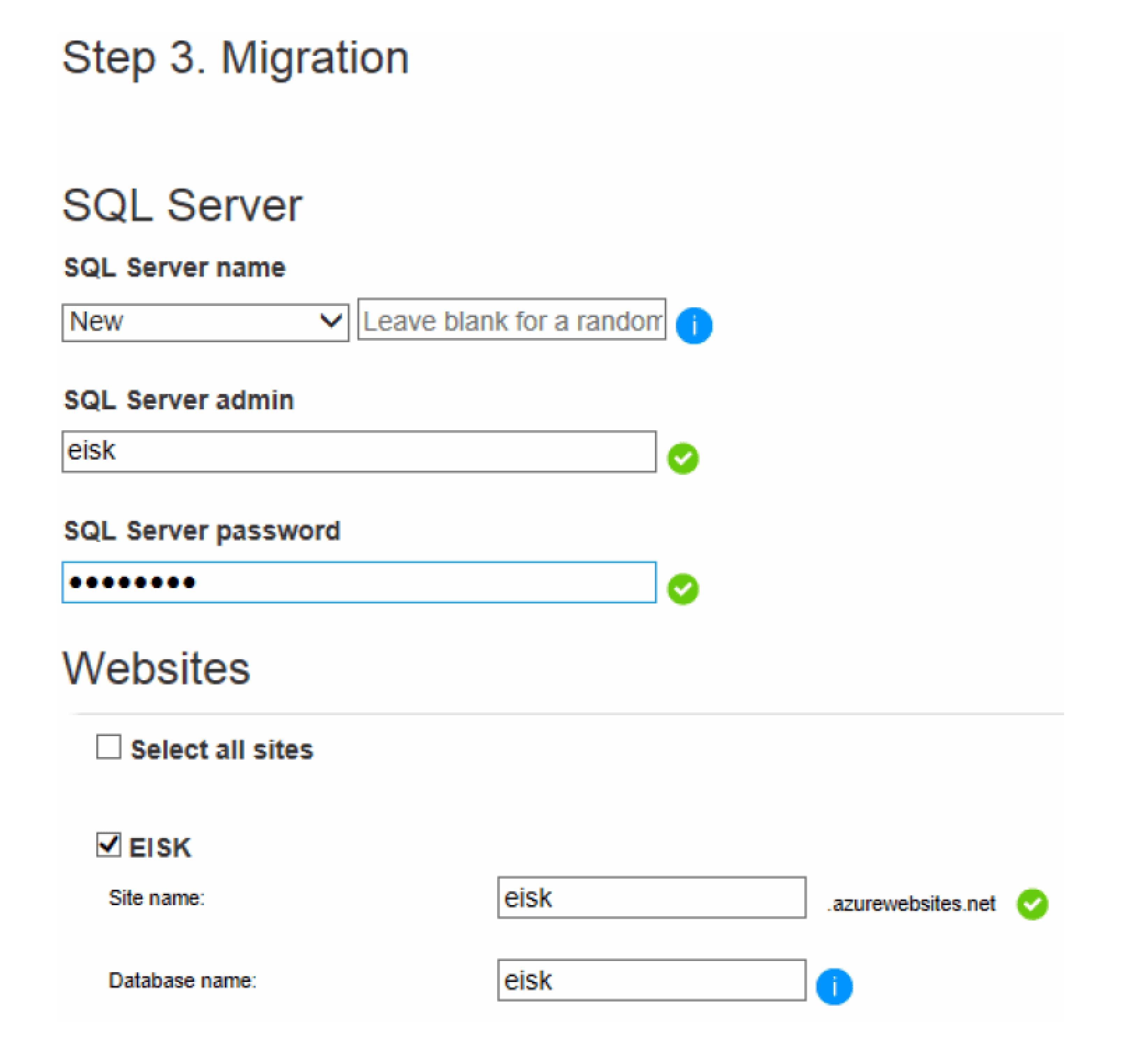


图 6-18　指定数据库和网站设置

如图 6-19 所示，在自定义设置中可以指定网站的模式和工作机的规格。

Customize Settings: EISK

Website

Site Mode

Standard

Worker Size

Small (1 core, 1.75 GB Memory)

Server farm

Create New

Enable Azure Active Directory

图 6-19　网站自定义设置

同样，也可以指定 SQL 服务器的自定义设置，包括版本、性能级别等。

现在，可以开始迁移网站和数据库了。如图 6-20 所示，该工具首先在指定的数据中心创建网站和 SQL Azure 数据库。

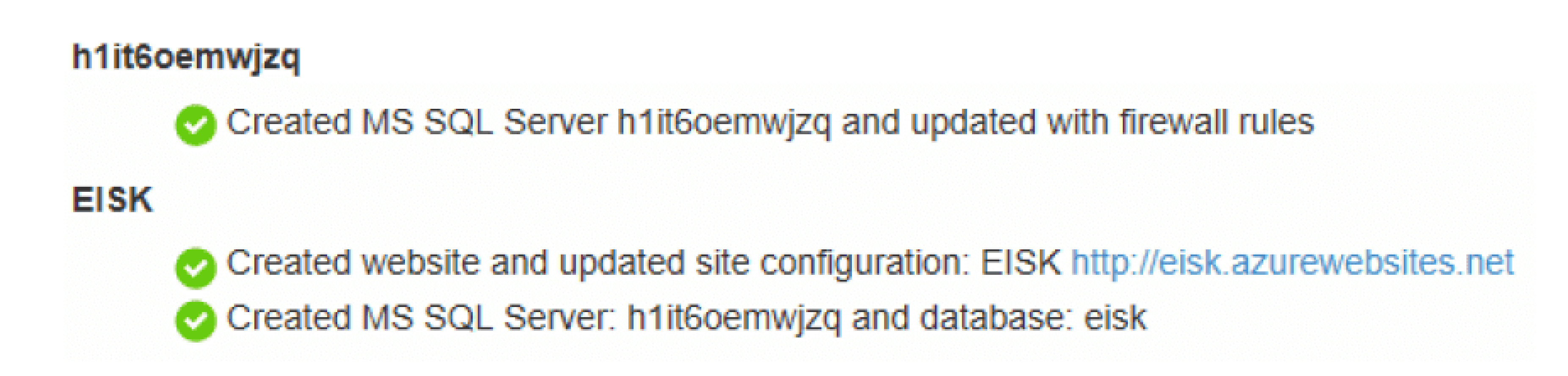


图 6-20 创建网站和数据库

现在，单击 Begin Publish，该工具将应用从本地部署到 Azure 网站，并将数据库迁移到 Azure 数据库。

通过该工具，只需几分钟就可以将运行在本地的网站和数据库迁移到 Azure 网站。

6.4 将 Azure 网站迁移到另一个数据中心

为了降低网络延迟，一般将网站部署在距离网站的客户最近的数据中心。比如澳大利亚、日本和新西兰的客户通常将网站部署在香港数据中心。为了进一步降低网络延迟、提高客户体验，微软公司持续在全球各地建设新的数据中心。比如，最近微软公司新建了 5 个数据中心（巴西 1 个，日本 2 个，澳大利亚 2 个）。很多澳大利亚和新西兰客户希望能够将部署在香港数据中心的网站迁移到新的澳大利亚数据中心。同样，日本的客户也希望将已有网站迁移到日本数据中心。对于生产网站而言，客户希望在迁移过程中网站仍然持续可用，不能够有宕机时间影响公司业务。

在本节，以 maws.azurewebsites.net 网站为例介绍如何不宕机将该网站从香港数据中心迁移到日本数据中心。

maws 网站是一个 ASP.NET 网站，后台采用 SQL 数据库。如图 6-21 所示，maws.azurewebsites.net 网站绑定有两个域名：modern.waws.cn 和 legacy.waws.cn。该网站位于东亚数据中心（香港）。

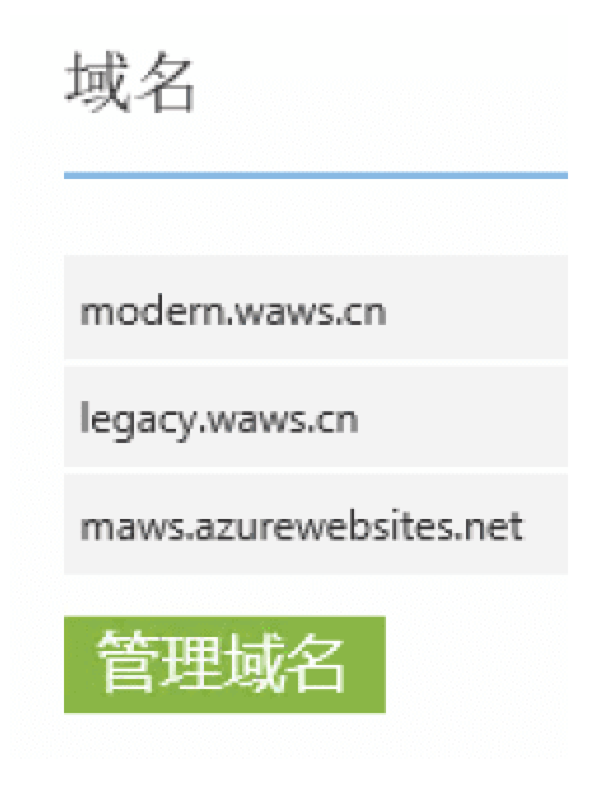


图 6-21 maws 网站

如图 6-22 所示，该网站显示通讯录信息，在页面底部显示网站名称。

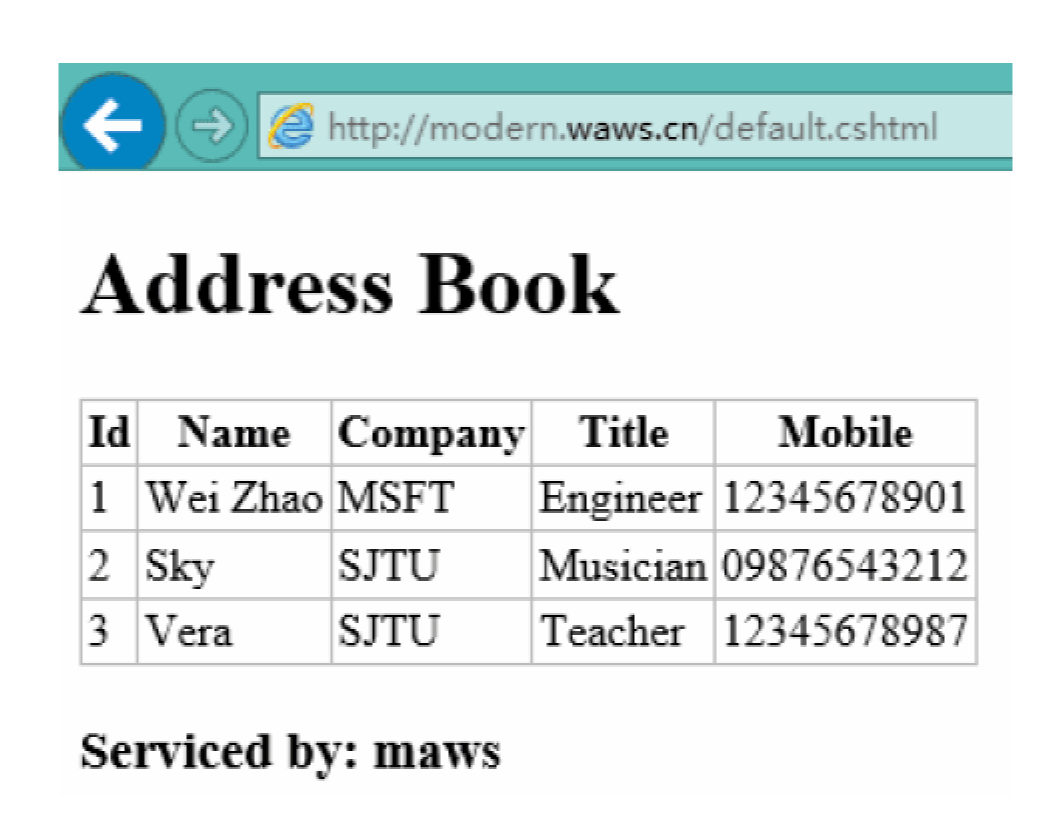


图 6-22　maws 网站

下面，介绍迁移网站的具体步骤。

6.4.1　备份当前网站

（1）登录到管理门户网站，在左侧导航栏选择“网站”。
（2）在网站列表中选择要备份的网站。
（3）在顶部菜单栏选择“备份”。
（4）如图 6-23 所示，指定存储备份的 Azure 存储账户，并选定要备份的数据库。



图 6-23　备份当前网站

（5）单击底部命令栏的“立即备份”。

6.4.2　创建新的网站

（1）登录到管理门户网站，单击左下角的新建。
（2）然后，选择“计算”→“网站”→“快速创建”。
（3）如图 6-24 所示，输入网站名称，选择“创建新的 WEB 宿主计划”，在本例中，要把 maws 网站迁移到日本西部数据中心，因此在区域一栏中选择“日本西部”。



图 6-24 创建新的网站

（4）网站创建完成后，在网站列表中选择该网站将其打开。在顶部导航菜单选择“缩放”，如图 6-25 所示，将“WEB 宿主计划模式”设置为标准模式。单击底部命令栏的“保存”按钮，保存修改。



图 6-25 网站升级为标准模式

6.4.3 将现有网站恢复到新建的网站

（1）登录到 Azure 管理门户网站，在左侧导航栏选择“网站”。

（2）在网站列表中选择要新建的网站，在本例中是 6.4.2 节创建的 maws-jpwest 网站。

（3）在顶部菜单栏选择“备份”。

（4）在底部命令栏选择“立即还原”。

（5）在“选择备份源”对话框中选择存储账户，单击浏览文件夹图标。

（6）如图 6-26 所示，选择存储备份的 Azure 存储账户，并选择网站备份文件。Azure 网站备份文件保存在名为 websitebackups 的容器中。备份文件命名方式为“网站名称_备份时间.zip”。

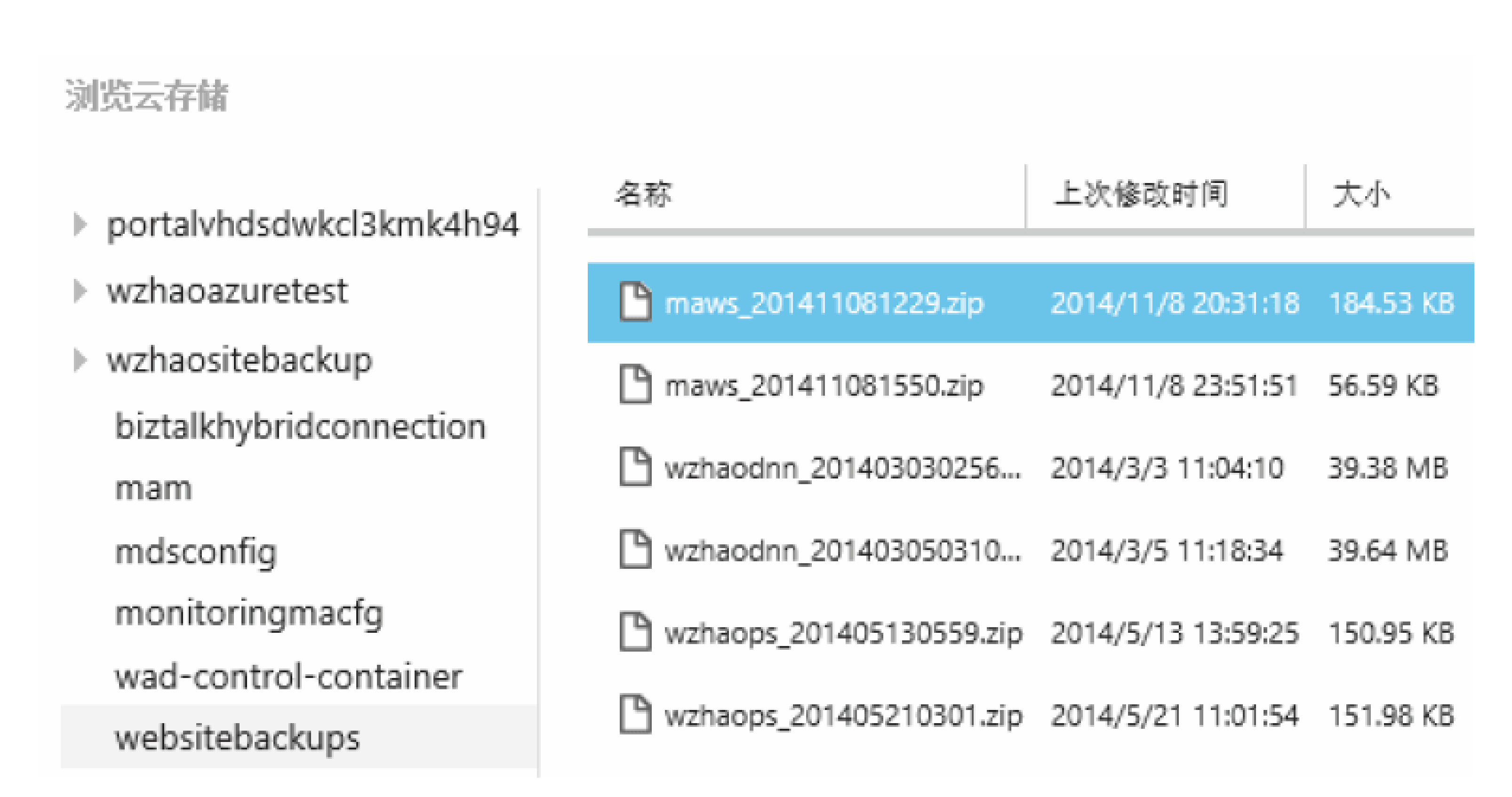


图 6-26 选择备份文件夹

（7）单击“下一步”按钮，如图 6-27 所示，选择“还原到当前网站”。关于数据库，可以选择将数据库还原到已有的数据库服务器，或者新建 SQL 数据库服务器。



图 6-27　网站还原设置

（8）在本例中，选择“新建 SQL 数据库服务器”，并选中“自动调整连接字符串”选项。然后单击“下一步”按钮。

（9）如图 6-28 所示，在“新建 SQL 数据库”对话框中，输入登录名和密码，并指定数据中心。基于性能考虑，网站必须和数据库在相同的数据中心。然后单击“完成”按钮，开始还原网站和数据库。



图 6-28　新建 SQL 数据库

6.4.4　验证新的网站

由于在还原网站时选择了自动调整数据库连接字符串，因此在网站还原后无须修改 web.config 中的字符串即可访问网站。这是因为在还原过程中，在网站的配置中自动添加了一个同名的连接字符串。在 4.6.2 节中，详细介绍了该功能。如图 6-29 所示，在 maws-jpwest 网站的配置页面，可以看到这个连接字符串。

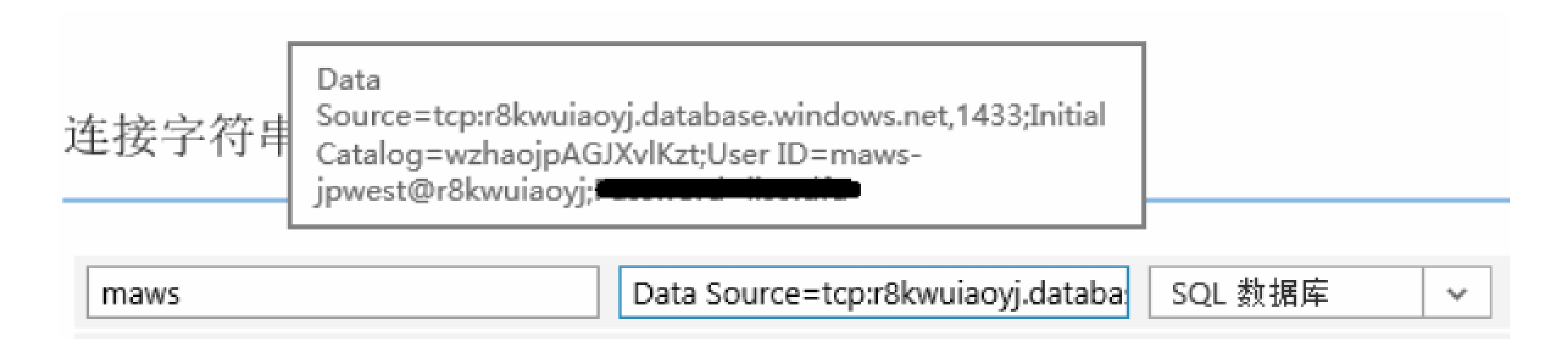
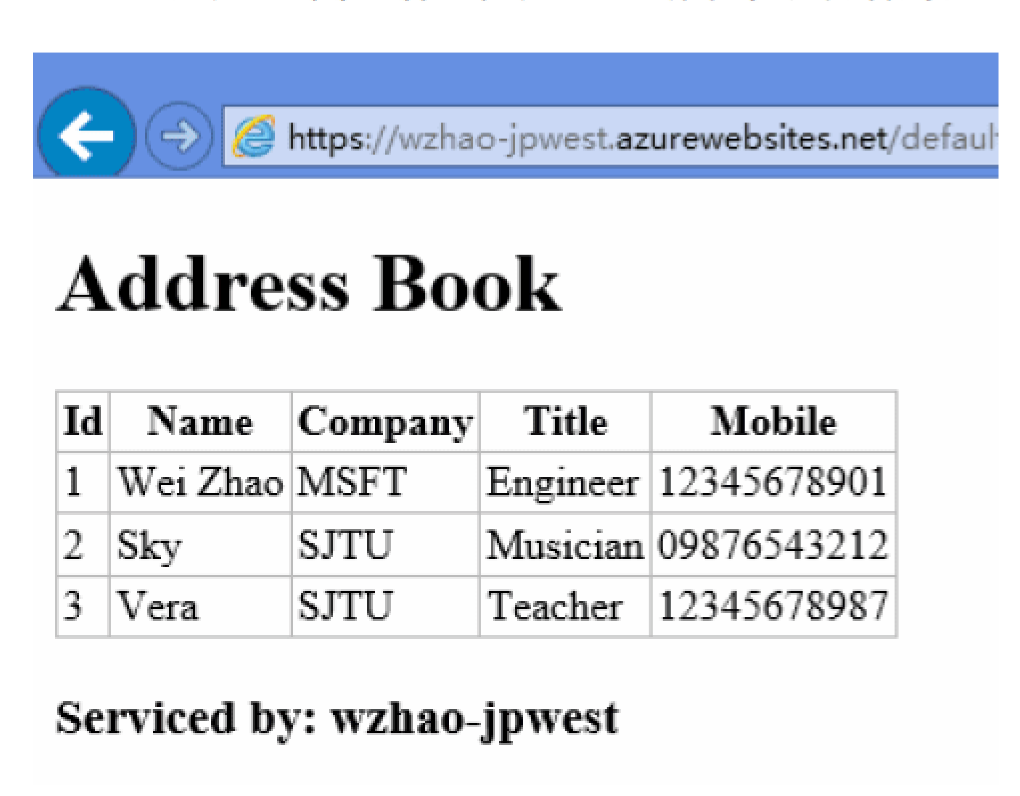


图 6-29　在网站还原过程中自动添加的连接字符串

现在，访问 maws-jpwest 网站，得到如图 6-30 所示的结果。



Id	Name	Company	Title	Mobile
1	Wei Zhao	MSFT	Engineer	12345678901
2	Sky	SJTU	Musician	09876543212
3	Vera	SJTU	Teacher	12345678987

图 6-30　新的网站访问结果

6.4.5　修改 DNS 配置

验证新的网站访问一切正常后，需要修改自有域名的 DNS 配置，将其指向新的网站。如图 6-31 所示，在网站还原过程中，网站绑定的自有域名都会被还原，

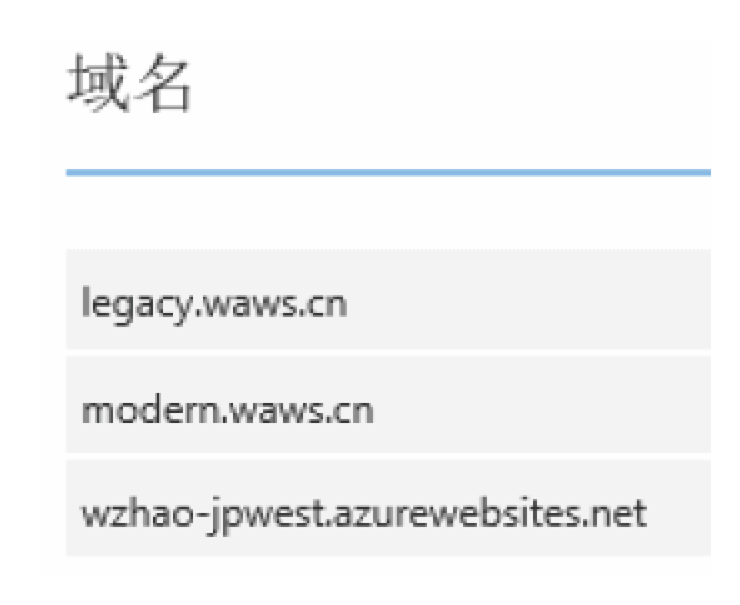


图 6-31　还原的自有域名

因此，无须再次重新绑定域名，只需要修改 DNS 配置，将自有域名指向新的网站即可。原有的 DNS 配置如表 6-2 所示。

表 6-2　原有的 DNS 配置

域　　名	记录类型	指　　向
modern.waws.cn	A	65.52.168.70（香港数据中心部署单元 IP 地址）
legacy.waws.cn	CNAME	maws.azurewebsites.net

修改后的 DNS 配置如表 6-3 所示。

表 6-3　修改后的 DNS 配置

域　　名	记录类型	指　　向
modern.waws.cn	A	138.91.16.18（日本西部数据中心部署单元 IP 地址）
legacy.waws.cn	CNAME	Maws-jpwest.azurewebsites.net

由于 DNS 解析结果有一定时间的缓存。当修改 DNS 后，如果客户使用的是本地 DNS 缓存，那么这些客户仍然会访问原有的网站。如图 6-32 所示，当 DNS 缓存失效后，客户端自动重新解析 DNS，之后客户访问的是新网站。

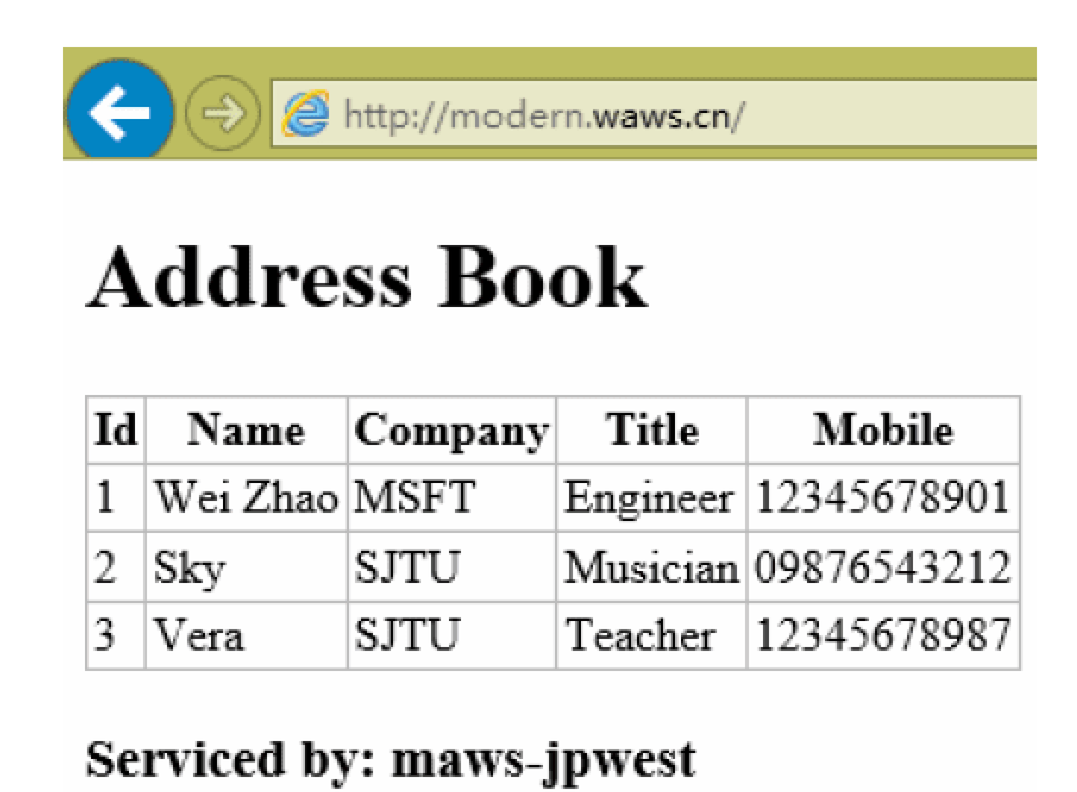


图 6-32　位于日本数据中心的新网站

此时，可以关闭旧的网站以节约成本。

6.5　参考文献与扩展阅读

1. Database Export and Import

MySQL 文档，介绍了如何通过 MySQL Workbench 导出和导入数据库。

http://dev.mysql.com/doc/workbench/en/wb-admin-export-import.html

2. 使用 Web Deploy 迁移 MySQL 数据库

Web Deploy 提供了 dbfullMysql 组件，通过该组件，Web Deploy 可以轻松地操作 MySQL

数据库，包括数据导入与导出。

http://blogs.iis.net/msdeploy/archive/2009/03/30/msdeploy-sample-custom-provider-to-sync-mysql-databases-dbfullmysql.aspx

3．WordPress 官网

WordPress 是一个开源的 Web 应用，是最受欢迎的 CMS 网站系统，可以用来创建个人网站和企业网站。它基于 PHP 开发，后台采用 MySQL 数据库（也可以采用 Microsoft SQL Server）。

http://wordpress.org/

4．NopCommerce 官网

NopCommerce 是一个开源的电子商务网站应用。它基于 ASP.NET MVC，后台采用 Microsoft SQL Server 数据库。

http://nopcommerce.codeplex.com/

5．BlogEngine.NET

BlogEngine.NET 可能是最轻量化的基于 ASP.NET 的博客网站系统。它的后台可以基于 XML 或者 Microsoft SQL Server。

http://blogengine.codeplex.com/

6．MySQL Workbench

MySQL Workbench 是管理 MySQL 的 UI 工具。它提供了数据导出和导入功能，这些功能基于 mysqldump.exe 工具。通过 MySQL Workbench 的数据导出和导入功能，可以轻松地将 MySQL 数据库迁移到 Microsoft Azure 上。

http://dev.mysql.com/doc/workbench/en/wb-mysql-connections-navigator-management-data-export.html

http://dev.mysql.com/doc/workbench/en/wb-mysql-connections-navigator-management-data-import-restore.html

7．Azure Websites Backups

Azure 网站备份与还原功能允许用户轻松地手动或者自动备份网站。

http://azure.microsoft.com/en-us/documentation/articles/web-sites-backup/

http://azure.microsoft.com/en-us/documentation/articles/web-sites-restore/

第 7 章　基于 Azure 网站构建高性能 Web 应用

Azure 网站提供可靠的企业级基础架构，并自动修补 Web 服务器和操作系统，从而使基础架构保持最新。这些基础架构支撑着世界上一些最大的企业的关键 Web 应用。Azure 网站自动对请求进行负载平衡，只需几秒钟即可纵向或横向扩展或自动缩放，可以轻松应对每天数百万次的请求，并且无比稳定。表 7-1 列出了关键的 Web 应用的要求和对应的 Azure 功能。

表 7-1　高性能 Web 应用关键要求

关键要求	对应的 Azure 功能
安全	Azure 网站集成 DDOS 防御功能
	IP 地址限制/动态 IP 地址限制功能
	SNI SSL/IP SSL
	集成 Azure AD 认证
	网站备份与恢复服务
全球网络	Azure Traffic Manager（流量管理器）
	Azure CDN（内容传输网络）
高性能	Azure 缓存服务
	横向扩展
	自动缩放
快速开发和部署	支持持续部署（TFS/Git）
	网站部署槽
	生产环境测试
	Web 作业（WebJobs）
	远程调试

在本章中，将具体讨论如何在 Azure 网站中集成其他 Azure 服务，构建/开发高性能、企业级的应用。

7.1　高性能 Azure 网站典型架构

图 7-1 是一个典型的基于 Azure 网站的高性能、高可靠性的架构。该架构通过集成其他 Microsoft Azure 服务，如流量管理器、内容传输网络、缓存服务等，构建可靠的高性能应用程序。

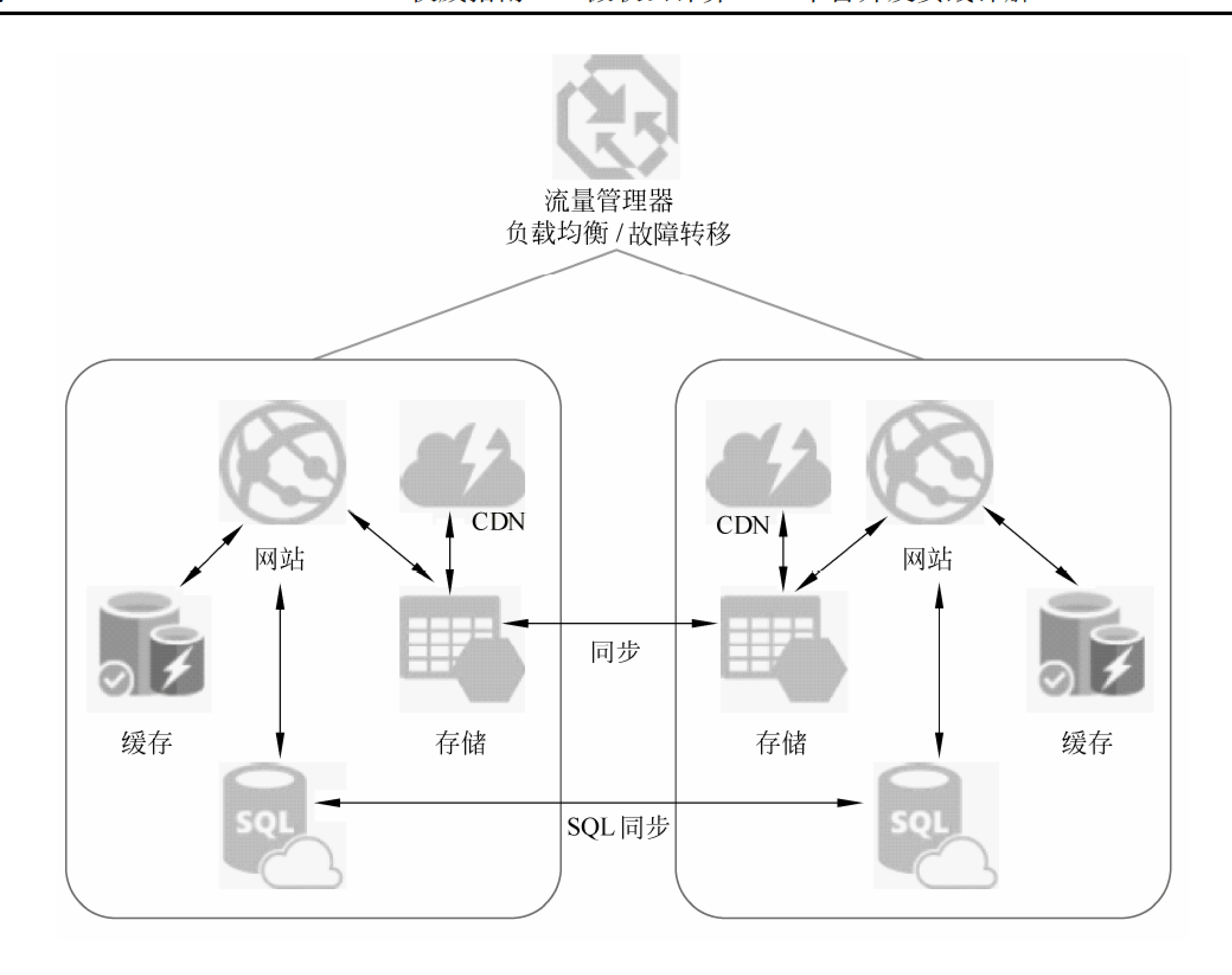


图 7-1　基于 Azure 网站的高性能应用架构

Azure 流量管理器允许将网站克隆并分布在不同的数据中心。客户请求自动被分发到距离自己最近的数据中心。Azure 流量管理器在提供负载均衡的同时提供高可靠性保障。当某个数据中心的网站出现问题时，Azure 流量管理器自动将客户请求分发到其他可用的数据中心，确保服务不受影响。

通过遍布全球的内容传输网络（CDN）节点，应用可以无限接近最终客户，进一步降低网络延迟。

Azure 缓存服务提供了高性能、可靠的缓存解决方案，Azure 网站可以利用 Azure 缓存服务提高性能。尤其是在突发高峰的情况下，Azure 缓存可以提高网站响应速度。

SQL Azure 提供了跨地域同步功能，确保数据一致的同时提供了灾难冗余功能。

图 7-2 在图 7-1 的基础上进一步集成了活动目录（Active Directory）和 VPN/混合连接/服务总线等服务，称为高性能、高可靠和安全的关键企业级 Web 应用的架构。

Azure 活动目录提供了 Azure 应用的身份管理和访问控制功能。Azure 活动目录可以轻松集成本地部署的 Windows 活动目录，实现应用程序的单一登录。可以利用它构建可靠、安全的新式业务应用程序。

通过混合连接、VPN 或者 Azure 服务总线，可以轻松地将运行在企业内部的应用与运行在 Azure 上的云端应用集成，实现混合云解决方案。

在本章，将逐个讨论如何在 Azure 网站中集成缓存服务、流量管理器、CDN、Azure 活动目录以及混合连接等功能。

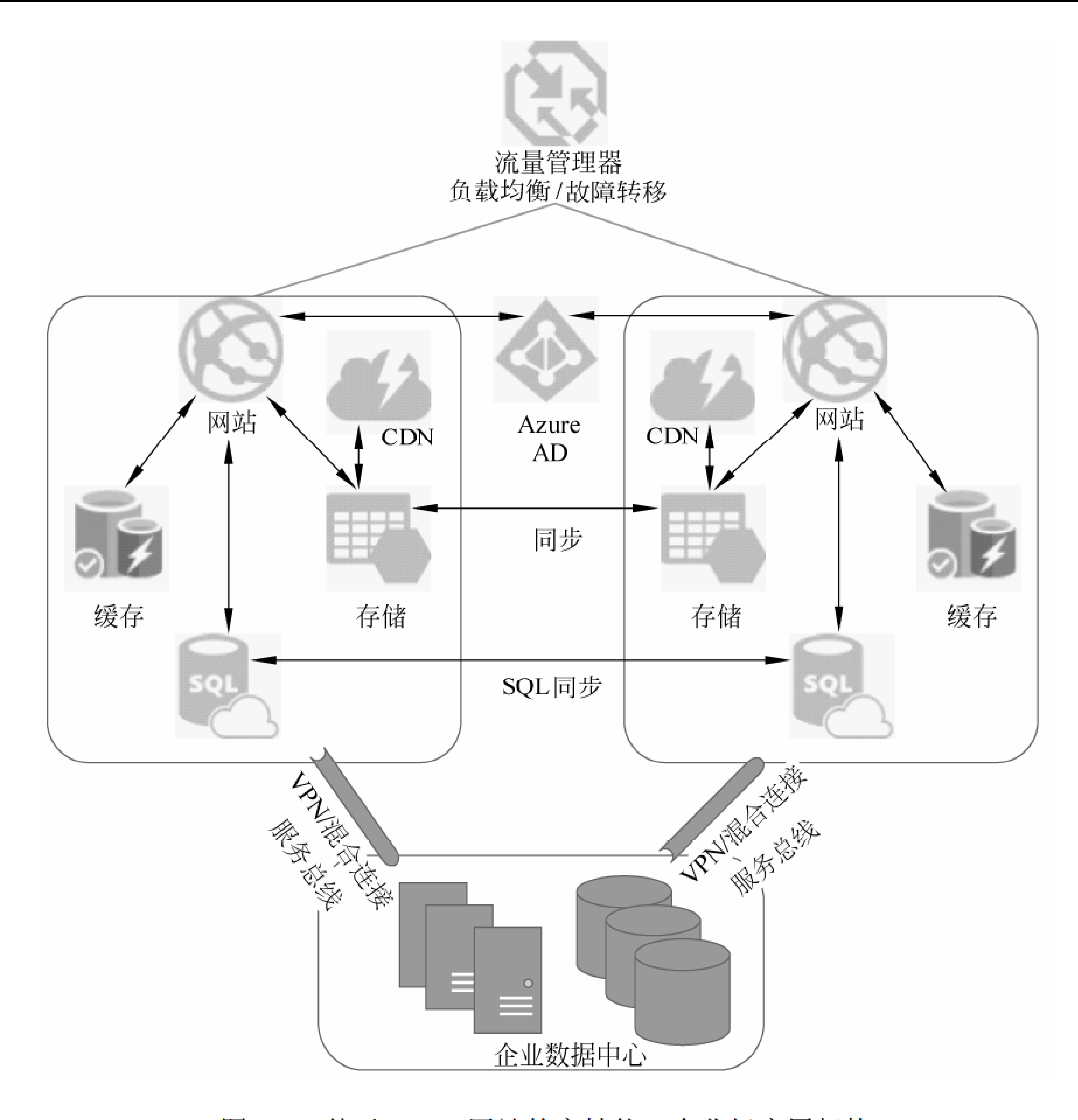


图 7-2　基于 Azure 网站的高性能、企业级应用架构

7.2　利用 Microsoft Azure 缓存服务（预览版）提高性能

7.2.1　Microsoft Azure 缓存服务（预览版）简介

Microsoft Azure 缓存服务（预览版）是全新的缓存解决方案。通过 Microsoft Azure 缓存服务，可以在 Microsoft Azure 中构建快速且可缩放的应用程序。可以在任何数据中心创建专用的、安全的缓存；可以全面控制该缓存，确保对业务关键数据进行隔离。

Microsoft Azure 缓存服务有助于应用程序即使在用户负载增加的情况下也能更加迅速地做出响应。单独的分布式缓存层可以更加高效地使用应用层的计算资源。

Microsoft Azure 缓存服务具有如下特点：

（1）支持多种 Azure 服务。Microsoft Azure 网站、Microsoft Azure 云服务和 Microsoft Azure 虚拟机都可以利用缓存的功能。

（2）跨应用程序共享数据。使用缓存可在运行于不同 Microsoft Azure 服务的松耦合型应用程序之间共享数据，也可在 Microsoft Azure 中同一应用程序的不同实例之间共享数据。

（3）轻松管理。缓存服务（预览版）可通过新增的 Microsoft Azure 管理门户网站轻松实现管理。可以轻松创建缓存、缩放缓存、配置缓存并监视缓存的运行状况和性能。

（4）没有配额限制。Microsoft Azure 缓存服务没有带宽和连接限制，物理容量是唯一的限制因素。用户只需要关注应用程序及其数据需求，根据缓存大小付费。

（5）隔离、灵活性和控制。Microsoft Azure 缓存服务提供专用的缓存，可对业务关键数据进行隔离。

- Memcache 兼容性

将使用 Memcache 的应用程序无缝迁移到缓存服务（预览版），代码不发生更改（注：Microsoft Azure 网站不支持 Memcache）。

7.2.1.1　缓存方案

Microsoft Azure 缓存服务提供了 3 种大小的缓存方案以满足不同应用的要求。

1．Basic Cache

Basic 方案提供 128MB～1GB 的缓存空间。在 Basic 模式下，多个客户的缓存服务共享相同的硬件设备。Basic 模式仅支持一个默认缓存，允许配置失效和驱逐策略。

2．Standard Cache

Standard 方案提供 1GB～10GB 的高速缓存方案。Standard 模式下，缓存服务被托管在专用硬件上，包含一个默认的缓存，并支持多达 10 个命名缓存。除了 Basic 方案的所有功能外，Standard 方案还支持通知。

3．Premium Cache

Premium 方案支持 5GB～150GB 的高速缓存。高级缓存被托管在专用硬件上，包含一个默认的缓存，并并支持多达 10 个命名缓存。除了 Standard 方案的所有功能外，Premium 方案还提供了高可用性支持。

7.2.1.2　失效策略

可以设置保存在 Azure 缓存中的对象的过期时间，默认的过期时间为十分钟。缓存的对象过期后，Azure 缓存服务自动从缓存中删除缓存对象。通过 API 将对象加入到缓存时，也可以指定对象的超时时间。

失效策略和过期时间一起用于决定缓存的对象何时过期。如表 7-2 所示，Azure 缓存服务提供了 3 种方式的失效策略。

表 7-2　缓存失效策略

失效策略	说　明
永不过期（never）	项目保留在缓存中，直到它们被驱逐。当指定为 Never 时，过期时间必须被设置为 0
绝对（absolute）	当一个项目被添加到缓存中后，开始计时，经过指定的时间间隔后，对象失效，从缓存中清除
滑动（sliding）	当一个对象被添加到缓存中后，开始计时。每当该对象被访问时，计时器归零重新开始计时。从最后一次访问开始，经过指定的过期时间后，对象失效，从缓存中清除。使用该策略，经常使用的对象将被缓存更长的时间

7.2.1.3　驱逐对象

当缓存的内存消耗接近选择的缓存方案所允许的最大值时，Azure 缓存将部分对象从内存中驱逐，无论它们是否已到期，直到内存压力被释放。Azure 缓存服务使用最近最少使用（LRU）方案选择被驱逐的对象。

如果禁止了驱逐功能，那么可能会遇到限流的问题。当缓存的内存消耗接近缓存方案的最大值时，Azure 缓存服务无法释放对象以解决该问题。在压力被释放之前，当试图添加新的对象时，会收到一个异常。

7.2.2　Azure 缓存服务应用架构

图 7-3 描述了 Microsoft Azure 缓存服务（预览版）的应用架构。Microsoft Azure 缓存服务是独立于应用的。利用 Azure 缓存服务可以在 Microsoft Azure 网站和云服务以及任何应用之间共享数据。

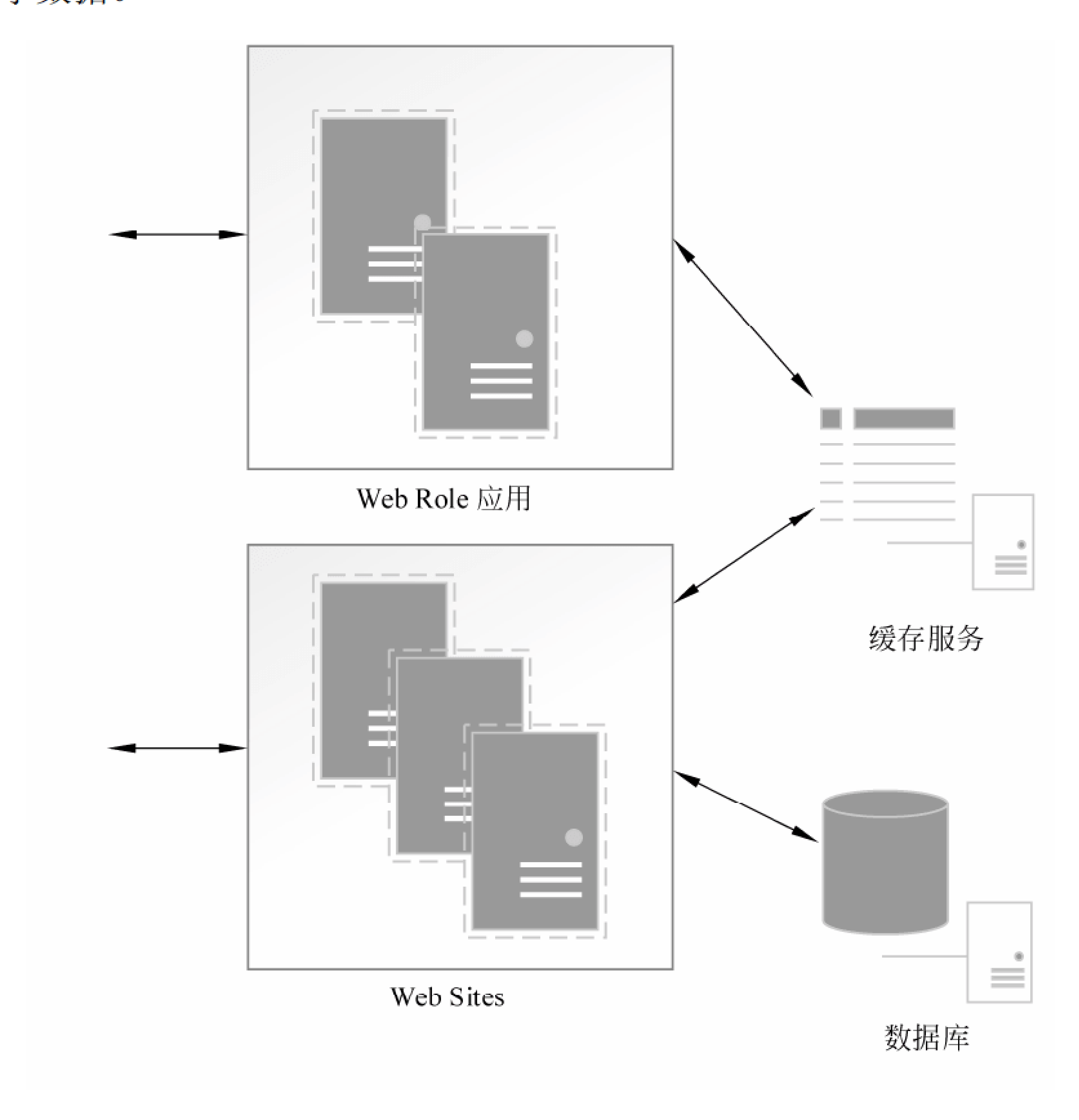


图 7-3　网站应用集成缓存服务架构

Microsoft Azure 缓存服务独立于应用本身，即使应用重启也不会影响缓存的信息。因此，不管是开发 Azure 云服务应用（Web Role）或是 Azure 网站应用，它都是存储会话的理想选择。关于如何使用 Azure 缓存服务存储会话，请参考下面的文档：

http://azure.microsoft.com/en-us/documentation/articles/web-sites-dotnet-session-state-caching/

本节以获取天气预报信息为例，详细讨论在 Microsoft Azure 网站中如何利用 Azure 缓存提高网站性能。下面的例子是访问一个公开的天气 API，获取上海市以及各区的当前天气情况，并将结果存储在 Azure 缓存中以提高性能。

7.2.2.1　创建缓存

（1）登录到 Microsoft Azure 管理门户网站。

（2）依次单击“新建”→“数据服务”→CACHE→QUICK CREATE。

（3）输入 ENDPOINT，并选择“区域”；根据应用的需要选择合适的缓存大小。

（4）单击 CREATE A NEW CACHE，创建缓存。

（5）缓存创建成功后，如图 7-4 所示，在 DANSHBOARD 页面中的“速览”区域，可以看到 ENDPOINT URL。之后，在通过 API 访问 Azure 缓存时，会用到 ENDPOINT URL。

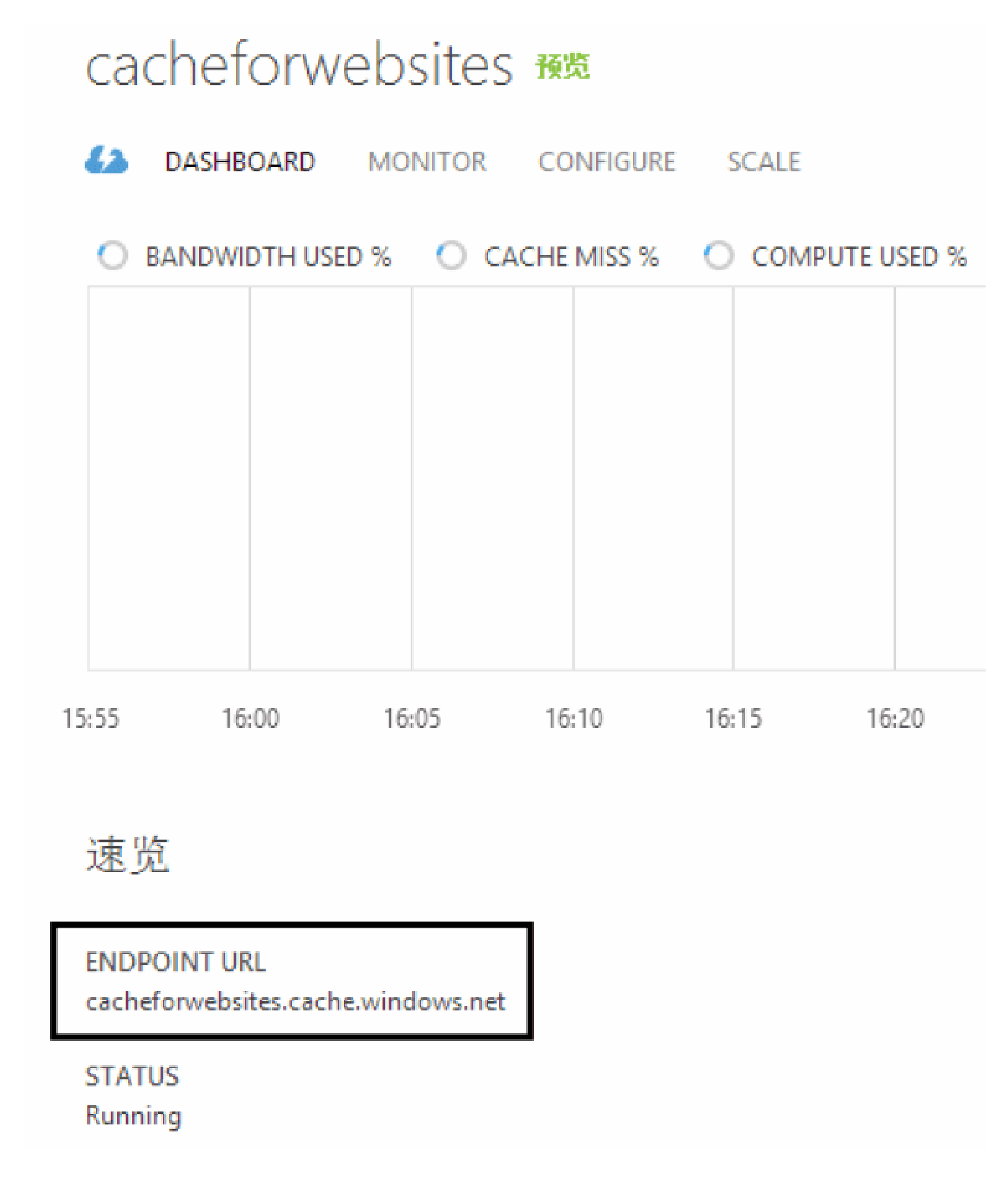


图 7-4　创建缓存

（6）获取访问密钥。与访问 Azure 存储相同，客户端访问 Azure 缓存时需要提供密钥进行认证。单击页面底部命令栏的 MANAGE KEYS 按钮，如图 7-5 所示，可以获取访问 Azure 缓存所需的密钥。之后，会用到该密钥。

（7）配置 Azure 缓存服务。如图 7-6 所示，可以单击页面顶部的 CONFIGURE，配置 Azure 缓存服务。默认失效策略是 Absolute，默认失效时间是 10 分钟。

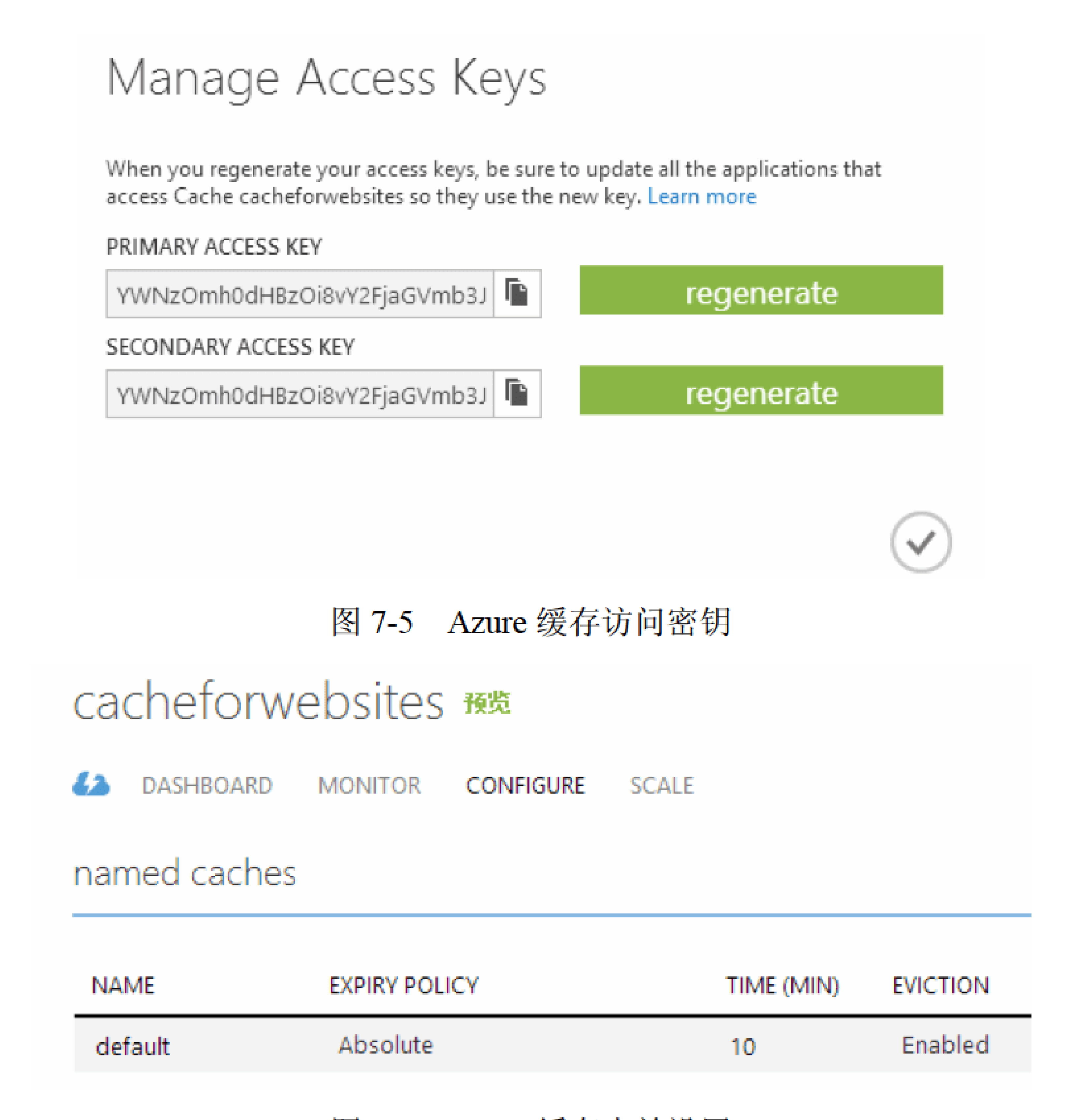


图 7-5　Azure 缓存访问密钥

图 7-6　Azure 缓存失效设置

7.2.2.2　在 Azure 网站中集成 Azure 缓存服务

（1）运行 Visual Studio 2013，选择“文件”→“新建”→“项目”命令。

（2）如图 7-7 所示，在“新建项目”窗口中选择 ASP.NET Web 应用程序，选择.NET Framework 4.5，命名为 UsingAzureCache，单击“确定”按钮。

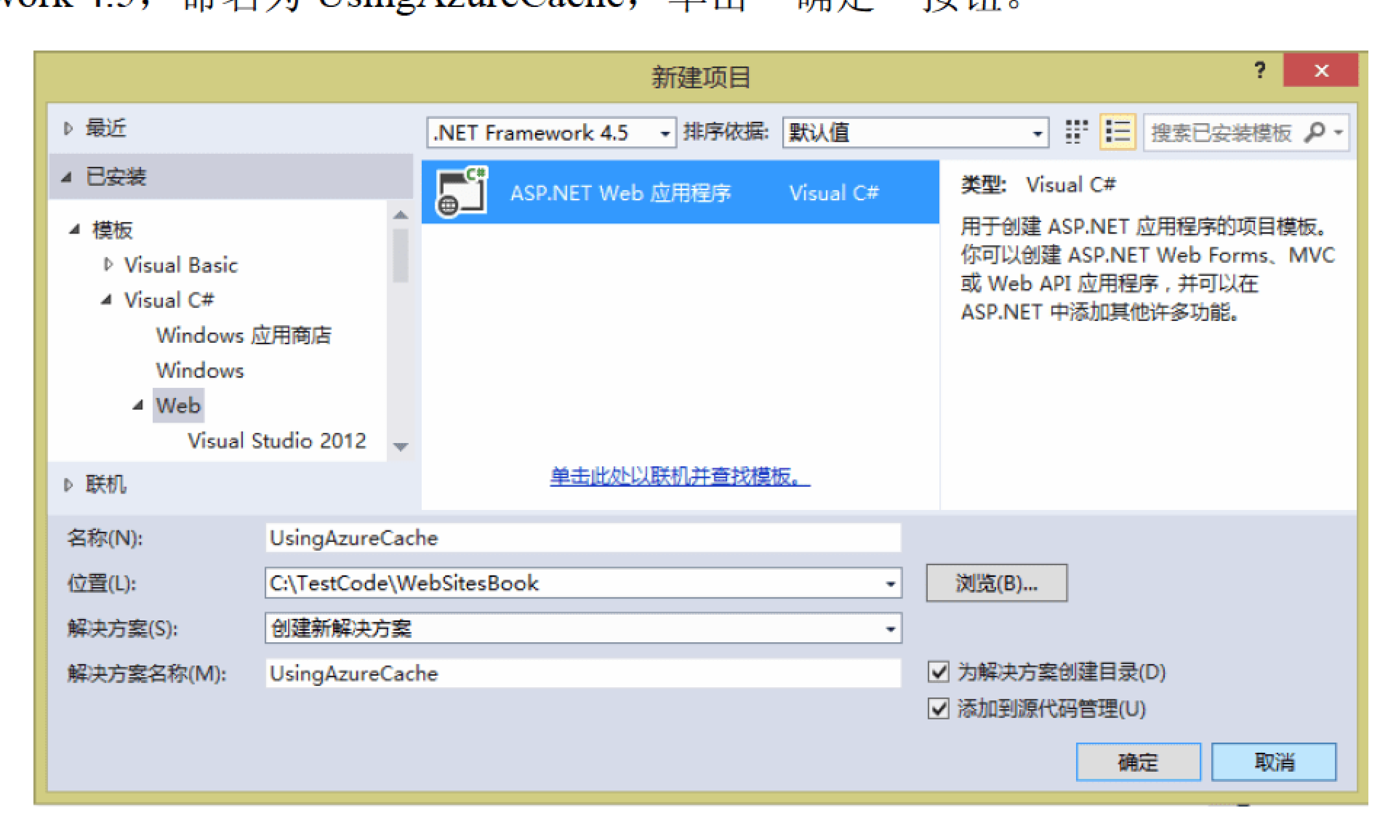


图 7-7　新建 ASP.NET 网站

（3）如图 7-8 所示，选择 Empty 模板，单击“确定”按钮，创建项目。

图 7-8 选择 Empty 模板

（4）安装 Microsoft.WindowsAzure.Caching 扩展包。在 Visual Studio 中选择“工具”菜单，选择“库程序包管理器”→“程序包管理控制台”。在控制台中运行下面的命令安装 Microsoft.WindowsAzure.Caching 扩展包：

```
Install-Package Microsoft.WindowsAzure.Caching
```

（5）配置 Microsoft Azure 缓存服务客户端。安装 Microsoft.Azure.Caching 扩展包时，在 Web.Config 中添加了<dataCacheClients>配置项，如下所示：

```
<dataCacheClients>
<dataCacheClient name="default">
  <!--To use the in-role flavor of Azure Caching,
       set identifier to be the cache cluster role name -->
  <!--To use the Azure Caching Service,
       set identifier to be the endpoint of the cache cluster -->
  <autoDiscover isEnabled="true" identifier="[Cache role name or Service
  Endpoint]" />
  <!--<localCache isEnabled="true" sync="TimeoutBased"objectCount=
  "100000" ttlValue="300" />-->
  <!--Use this section to specify security settings for connecting to your
  cache.
       This section is not required if your cache is hosted on a role that
       is a part of your cloud service. -->
    <!--<securityProperties mode="Message" sslEnabled="false">
```

```
        <messageSecurity authorizationInfo="[Authentication Key]" />
      </securityProperties>-->
    </dataCacheClient>
    </dataCacheClients>
```

在创建 Azure 缓存服务时，特别提到 ENDPOINT URL 和访问密钥。在上面的配置中，需要指定 ENDPOINT URL 给 identifier 配置项；指定访问密钥给 authorizationInfo 配置项。

（6）安装 NewtonSoft.Json 扩展包。

在程序包管理控制台中运行下面的命令安装 Newtonsoft.Json 扩展包：

```
Install-Package NewtonSoft.Json
```

（7）添加 System.Net.Http 引用。在解决方案资源管理器中，右击“引用”，选择“添加引用”命令。如图 7-9 所示，在左侧列表中选择“框架”，在中间 Assembly 列表中选中 System.Net.Http，单击“确定”按钮。

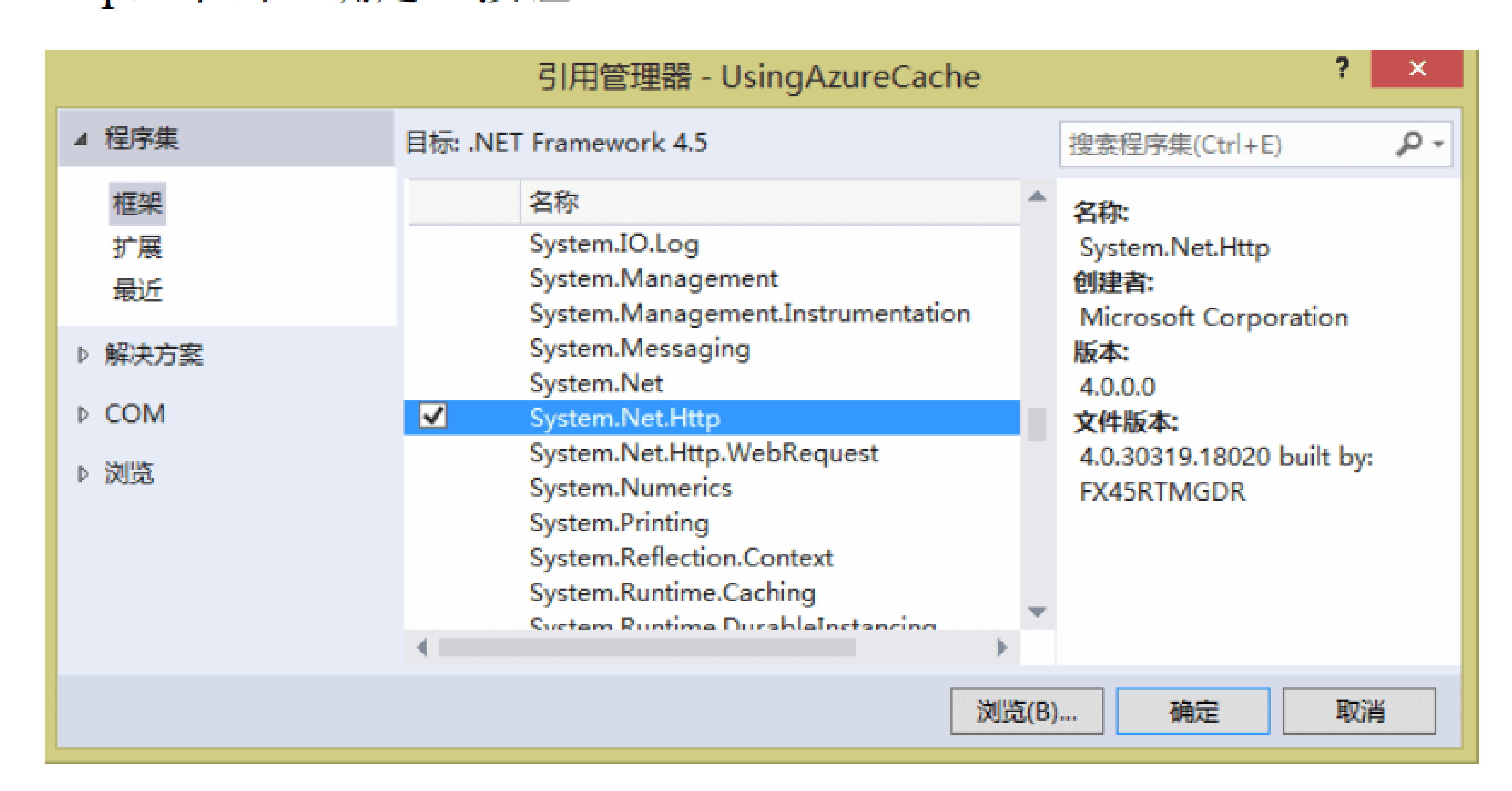


图 7-9　引用 System.Net.Http

（8）初始化缓存服务客户端。在 Global.asax.cs 文件中，添加如下代码初始化 Azure 缓存服务客户端。在之后的代码中，将使用 Global.defaultCache 来访问 Azure 缓存服务。

```
public class Global : System.Web.HttpApplication
{
    public static DataCacheFactoryConfiguration config;
    public static DataCacheFactory cacheFactory;
    public static DataCache defaultCache;
    protected void Application Start(object sender, EventArgs e)
    {
         // Cache client configured by settings in application configuration
         file.
        config = new DataCacheFactoryConfiguration("default");
        cacheFactory = new DataCacheFactory(config);
        defaultCache = cacheFactory.GetDefaultCache();
    }
```

```
    }
```

（9）在解决方案资源管理器中右击 UsingAzureCache 项目，选择“添加”→“新建项”命令，如图 7-10 所示，选择“类”，命名为 WeatherHelper.cs。

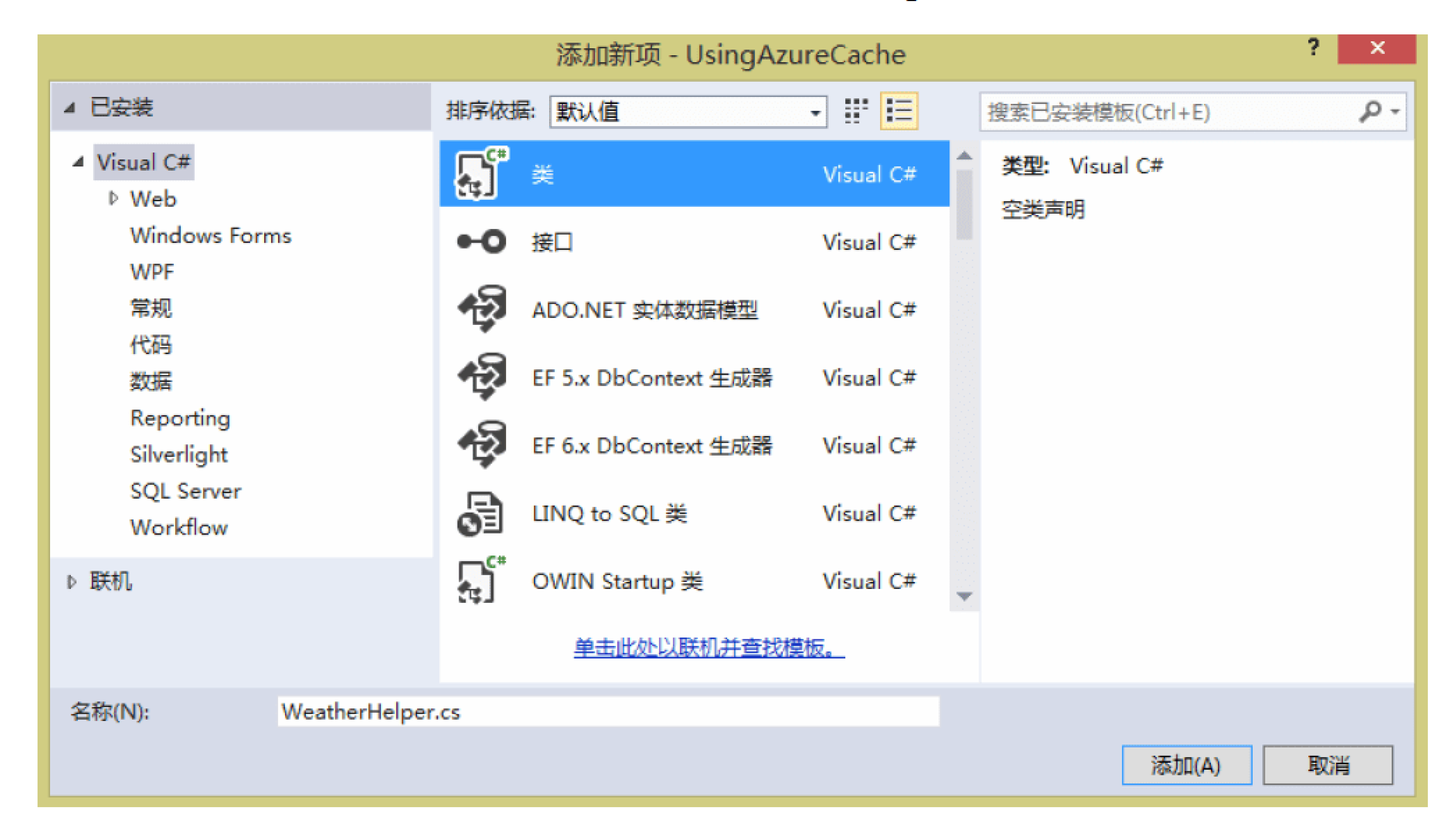


图 7-10　新建 C#类

（10）在该类中添加如下代码。下面的代码使用 HttpClient 异步调用天气 API 获取上海市和各个区的天气情况，并将其保存在一个 List 中返回。

```
public class WeatherHelper
{
    public class WeatherInfo
    {
        public string city { get; set; }
        public string cityid { get; set; }
        public string temp1 { get; set; }
        public string temp2 { get; set; }
        public string weather { get; set; }
        public string ptime { get; set; }
    }
    static string[] districts = { "101020100", "101020300", "101020500",
                                  "101020600", "101021300", "101020800",
                                  "101020900", "101021000", "101021100",
                                  "101021200", "101020200", "101020700" };
    static string url = @"http://www.weather.com.cn/data/cityinfo/
    {0}.html";

    public static async Task<List<WeatherInfo>> getWeather()
    {
```

```
            HttpClient client = new HttpClient();
            List<WeatherInfo> result = new List<WeatherInfo>();
            foreach (var dist in districts)
            {
                string distUrl = string.Format(url, dist);
                // Send a request asynchronously continue when complete
                HttpResponseMessage response = await client.GetAsync(distUrl);
                // Check that response was successful or throw exception
                response.EnsureSuccessStatusCode();
                // Read response asynchronously as JsonValue
                string content = await response.Content.ReadAsStringAsync();
                JObject jObj = JObject.Parse(content);
                WeatherInfo weather = new WeatherInfo
                {
                    city = (string)jObj["weatherinfo"]["city"],
                    cityid = (string)jObj["weatherinfo"]["cityid"],
                    temp1 = (string)jObj["weatherinfo"]["temp1"],
                    temp2 = (string)jObj["weatherinfo"]["temp2"],
                    weather = (string)jObj["weatherinfo"]["weather"],
                    ptime = (string)jObj["weatherinfo"]["ptime"]
                };
                result.Add(weather);
            }
            return result;
        }
    }
```

（11）在解决方案资源管理器中右击 UsingAzureCache 项目，选择“添加”→“新建项”命令，如图 7-11 所示，选择“Web 窗体”，命名为 default.aspx。

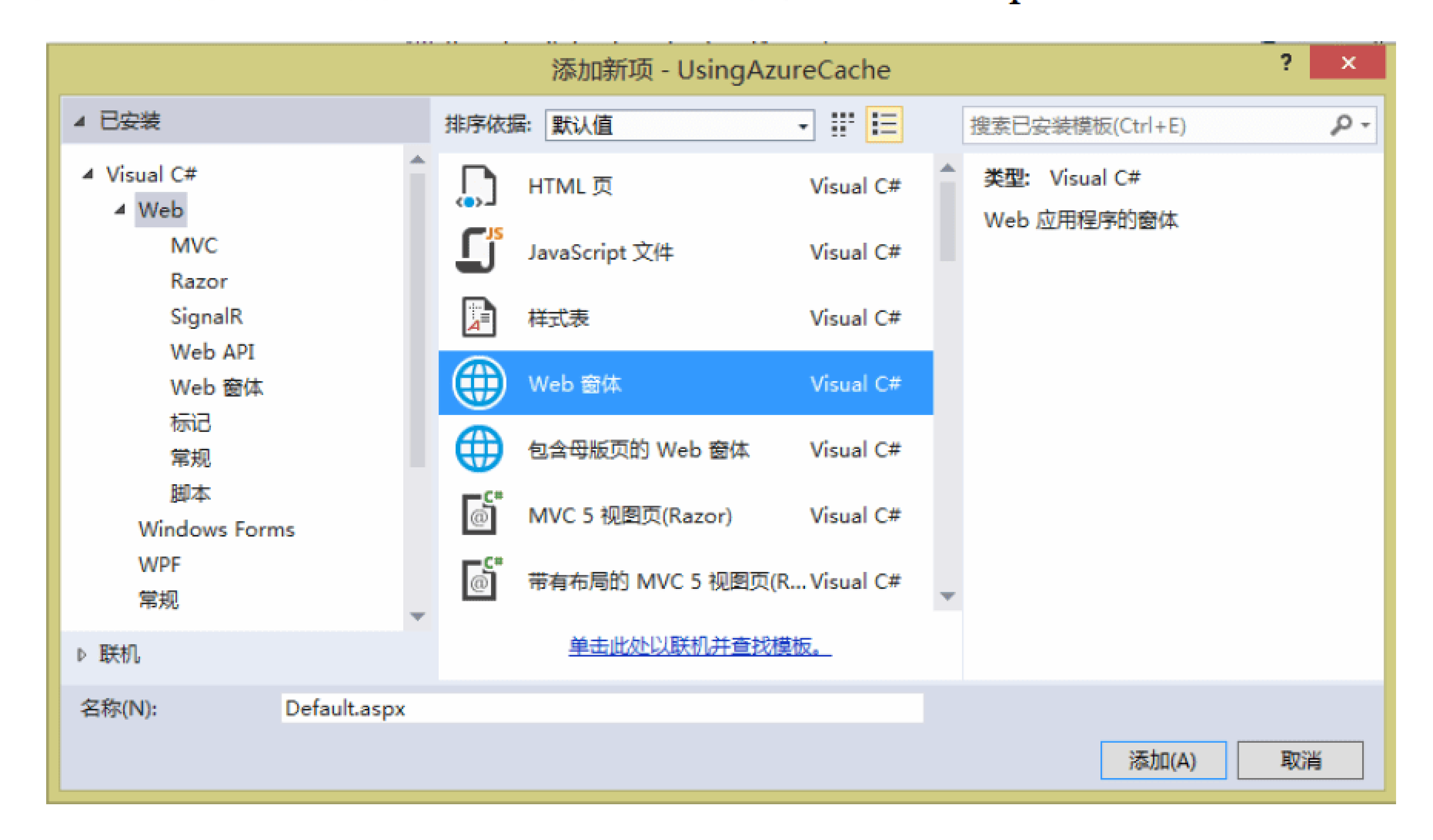


图 7-11　新建 Web 窗体

（12）打开 Default.axpx，在<body>中加入一个 GridView 对象用于显示结果。

```
<body>
    <form id="form1" runat="server" style="width:600px;margin:0 auto">
    <div>
        <h1>Shanghai Weather</h1>
         <asp:Label ID="LabelHeaderInfo" runat="server" />
        <br />
    </div>

    </div>
        <asp:GridView ID="GridViewResult" runat="server">
        </asp:GridView>
    </form>
</body>
```

（13）在 Default.aspx 文件中，找到 Page_Load 函数，添加查询并显示天气预报结果的代码。下面的代码首先在 Azure 缓存中查找天气预报结果，如果没有找到，异步调用 WeatherHelper 中提供的方法，通过天气 API 来查询天气情况，并将查询结果保存到 Azure 缓存中。

```
public partial class Default : System.Web.UI.Page
{

    List<UsingAzureCache.WeatherHelper.WeatherInfo> result = null;
    object cachedResult = null;
    protected async void Page Load(object sender, EventArgs e)
    {
        cachedResult = Global.defaultCache.Get("SHWeather");
        if (cachedResult == null)
        {
            result = await WeatherHelper.getWeather();
            Global.defaultCache.Put("SHWeather", result);
            LabelHeaderInfo.Text = "Serviced by "
            + System.Diagnostics.Process.GetCurrentProcess().Id.ToString()
            + " by calling weather API" + "<br />";
        }
        else
        {
            result = (List<UsingAzureCache.WeatherHelper.WeatherInfo>)
            cachedResult;
            LabelHeaderInfo.Text = "Serviced by "
                + System.Diagnostics.Process.GetCurrentProcess().Id.
                ToString()
                + " by Azure Cache Service" + "<br />";
        }
```

```
        var items = from n in result select n;

        var itm = result.Select(p => new { p.city, p.temp1 });

        GridViewResult.DataSource = items;
        GridViewResult.DataBind();
    }
}
```

（14）将应用部署到 Azure 网站后，第一次访问时，如图 7-12 所示，会看到天气信息是调用天气 API 获取的。

Shanghai Weather

Serviced by 43976 from weather API

city	cityid	temp1	temp2	weather	ptime
上海	101020100	16℃	22℃	中雨转小雨	18:00
宝山	101020300	16℃	22℃	中雨转小雨	18:00
嘉定	101020500	16℃	22℃	中雨转小雨	18:00
浦东南汇	101020600	16℃	21℃	中雨转小雨	18:00
浦东	101021300	16℃	22℃	中雨转小雨	18:00
青浦	101020800	16℃	22℃	小雨	18:00
松江	101020900	16℃	22℃	小雨	18:00
奉贤	101021000	16℃	22℃	中雨转小雨	18:00
崇明	101021100	16℃	22℃	中雨转小雨	18:00
徐家汇	101021200	16℃	22℃	中雨转小雨	18:00
闵行	101020200	16℃	22℃	小雨	18:00
金山	101020700	16℃	22℃	中雨转小雨	18:00

图 7-12　天气信息

（15）再次访问，如图 7-13 所示，会看到天气信息是从 Azure 缓存中获取的。

Shanghai Weather

Serviced by 43976 from Azure Cache Service

city	cityid	temp1	temp2	weather	ptime
上海	101020100	16℃	22℃	中雨转小雨	18:00
宝山	101020300	16℃	22℃	中雨转小雨	18:00
嘉定	101020500	16℃	22℃	中雨转小雨	18:00
浦东南汇	101020600	16℃	21℃	中雨转小雨	18:00
浦东	101021300	16℃	22℃	中雨转小雨	18:00
青浦	101020800	16℃	22℃	小雨	18:00
松江	101020900	16℃	22℃	小雨	18:00
奉贤	101021000	16℃	22℃	中雨转小雨	18:00
崇明	101021100	16℃	22℃	中雨转小雨	18:00
徐家汇	101021200	16℃	22℃	中雨转小雨	18:00
闵行	101020200	16℃	22℃	小雨	18:00
金山	101020700	16℃	22℃	中雨转小雨	18:00

图 7-13　从 Azure 缓存中获取的天气信息

（16）重启网站，会看到新的 Process ID，但是天气信息仍然是从 Azure 缓存中获取的。这表明，网站应用缓存的内容不受网站重启的影响。

（17）等待十几分钟后，再次访问，会发现应用再次调用天气 API 而不是从 Azure 缓存中获取天气预报。这是因为缓存失效后，相应的对象已经被从缓存中清除。

7.3 集成 Microsoft Azure 流量管理器提高性能与可靠性

7.3.1 Microsoft Azure 流量管理器简介

Microsoft Azure 流量管理器允许控制如何将用户的请求分发到指定的服务端点。Microsoft Azure 流量管理器支持云服务和网站。流量管理器的工作原理是将智能策略引擎应用到域名（DNS）查询中。使用 Microsoft Azure 流量管理器，Azure 网站可以分布在世界各地不同的 Microsoft Azure 数据中心。

流量管理器可以帮助用户实现以下目标：

（1）提高关键应用的可用性。流量管理器允许用户监控服务端点，并提供自动故障转移功能，提高关键应用的可用性。

（2）提高高性能应用程序的响应能力。可以将网站分布到位于世界各地的数据中心。流量管理器可以根据客户的地理位置，将客户请求转发到距离客户最近的数据中心，通过降低网络延迟改善网站的响应时间。

（3）零停机时间。可以离线升级或维护某个服务端点。流量管理器自动将用户请求转发到其他的服务端点，这样就可以不停机维护或者升级站点。

图 7-14 描述了配置了客户域名情况下流量管理器与 Azure 网站集成的工作原理。

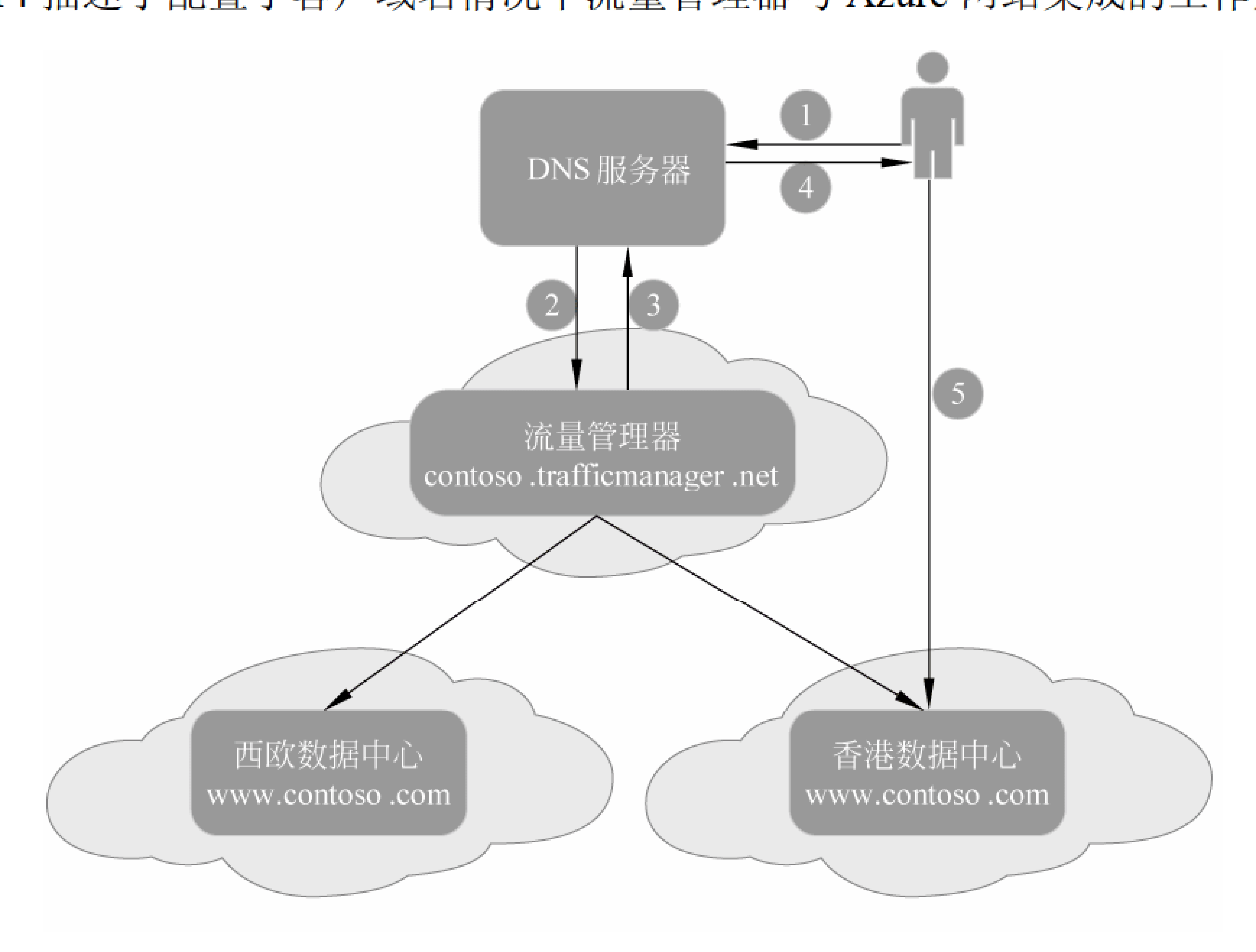


图 7-14 流量管理器工作原理

（1）当客户访问网站时，首先发送 DNS 请求解析域名 www.contoso.com。

（2）www.contoso.com 被解析为 contoso.trafficmanager.net。

（3）流量管理器根据配置的负载均衡方法和服务端点的监控结果选择一个网站端点。假如流量管理器选择了位于香港数据中心的网站，则返回香港数据中心网站的 IP 地址。

（4）DNS 解析结果返回给客户。

（5）客户端收到 DNS 解析结果，HTTP 请求直接发往香港数据中心网站，不经过流量管理器。

与网络负载均衡方法不一样，流量管理器只负责 DNS 解析，客户的 HTTP 请求并不通过流量管理器。简单来讲，流量管理器通过控制 DNS 解析来决定客户应该访问的网站。

DNS TTL 决定客户端的 DNS 解析结果缓存的时间。在 DNS 缓存失效之前，客户端不会再次发送 DNS 解析请求。因此，在此期间内客户的所有请求都将被转发至同一个端点，直到本地 DNS 缓存失效。当 DNS 缓存失效后，客户端重新发送 DNS 解析请求。

7.3.2　流量管理器负载均衡策略

Azure 流量管理器支持 3 种负载均衡方式。每个流量管理器配置文件在同一时间只能使用一种负载均衡的方法。当然，可以在任何时候改变负载均衡方法。

要注意的是，所有的负载均衡方法都依赖于监控。根据应用需要，指定最适合需求的负载均衡方法，并配置对应的监控设置。如果监控配置正确，流量管理器将监控端点的状态，包括云服务和网站。当流量管理器监控到某个端点不可用时，它自动将客户请求分流到其他服务端点。

流量管理器提供的 3 种负载均衡方法是故障转移、循环和性能策略。

7.3.2.1　故障转移（Failover）

通常通过为关键网站提供备份网站的方位提高服务的可靠性。这种情况下，所有客户请求都由首选的网站处理。当首选网站不可用时，客户请求将由备用网站处理。

Azure 流量管理器允许配置多个端点，并对多个端点进行排序。所有客户请求都会被转发到优先级最高的网站。当优先级最高的网站不可用时，所有的客户请求被转发到排在第二位的网站。如果排在第二位的网站不可用，则将用户请求转发到排在第三位的网站，以此类推。对于故障转移方案，选择端点的顺序很重要。

图 7-15 描述了故障转移方式的工作原理。流量管理器维护一个网站端点优先级列表。当优先级最高的香港数据中心网站不可用时，流量管理器选择第二优先级网站——西欧数据中心网站处理客户请求。

（1）当客户访问网站时，首先发送 DNS 请求解析域名 www.contoso.com。

（2）www.contoso.com 被解析为 contoso.trafficmanager.net。

（3）流量管理器总是返回香港数据中心网站。但是，当流量管理器发现香港数据中心网站已经不可用时，就会返回西欧数据中心的网站。

（4）DNS 解析结果返回给客户。

（5）客户 DNS 解析结果，HTTP 请求直接发往西欧数据中心网站，不经过流量管理器。

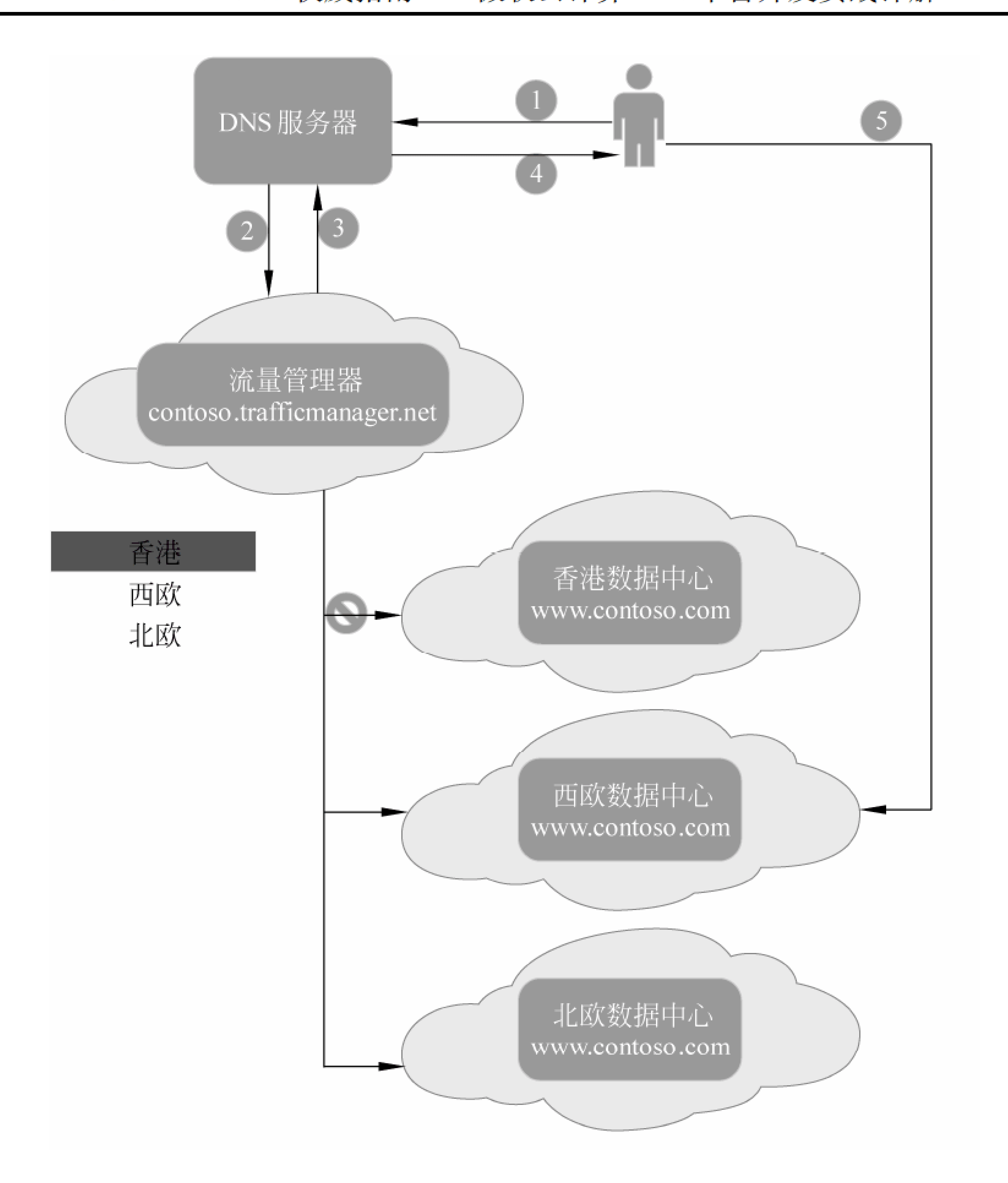


图 7-15 故障转移方法工作原理

7.3.2.2 循环（Round Robin）

循环方式下，流量管理器轮流返回配置的端点。图 7-16 描述了循环方式下流量管理器与 Azure 网站集成的工作原理。流量管理器维护端点列表，并记录上一个返回的端点。

（1）当客户访问网站时，首先发送 DNS 请求解析域名 www.contoso.com。

（2）www.contoso.com 被解析为 contoso.trafficmanager.net。

（3）因为上一个返回的端点是香港数据中心网站，因此流量管理器返回西欧数据中心网站。

（4）DNS 解析结果返回给客户。

（5）客户端收到 DNS 解析结果，HTTP 请求直接发往西欧数据中心网站，不经过流量管理器。

7.3.2.3 性能（Performance）

采用性能策略时，Azure 流量管理器将客户请求转发至网络延迟最小的端点。通常，“延迟最小”端点直接对应于最短的地理距离。性能负载平衡方法将允许基于位置和网络延迟进行负载均衡。但是该方式并没有考虑到网络配置或负载的实时变化。

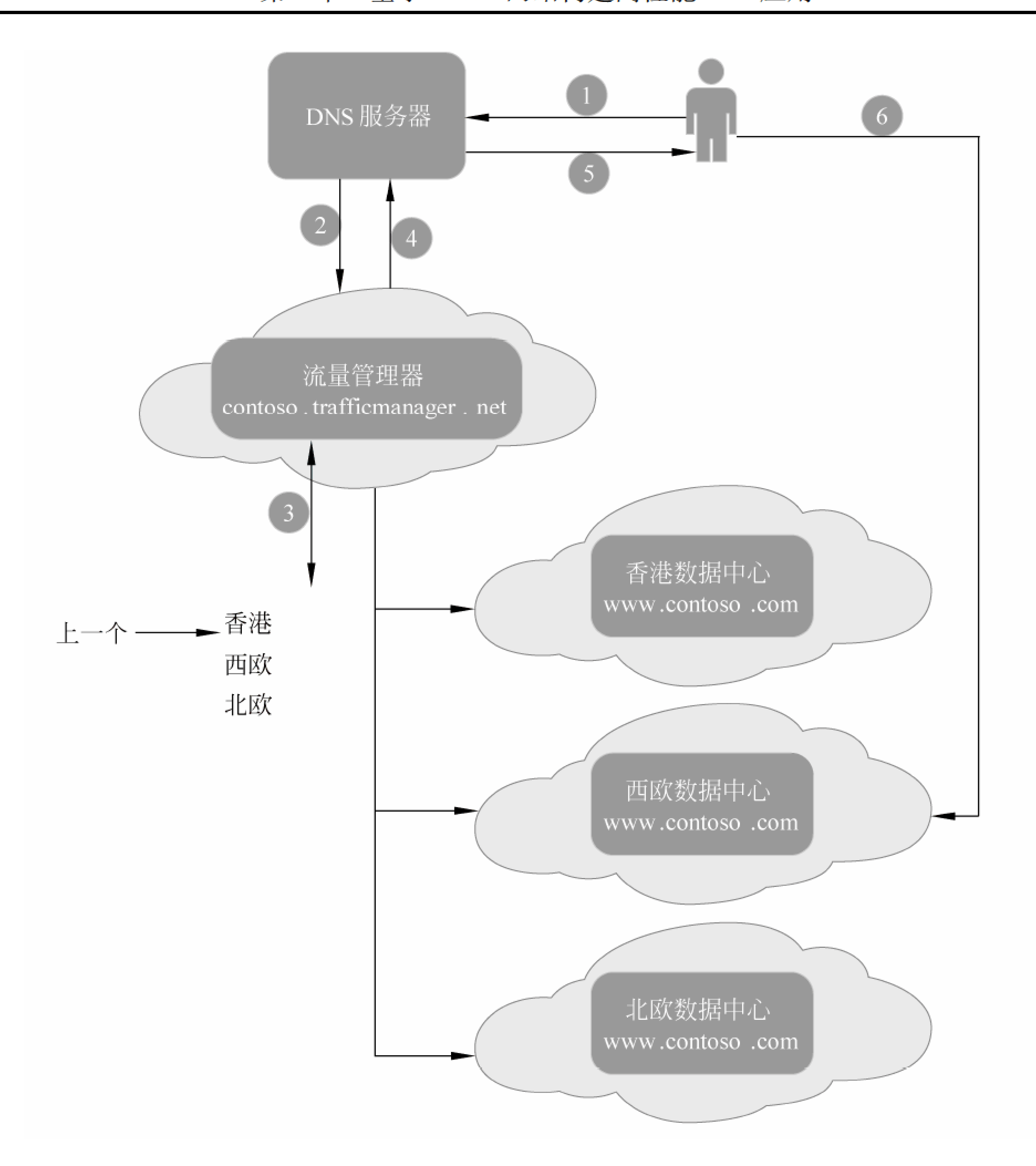


图 7-16　循环方式工作原理

性能负载均衡方法根据客户的地址，将客户的请求转发至离客户最近的端点。“最近”是根据不同 IP 地址与每个 Microsoft Azure 数据中心的网络延迟决定的。Azure 流量管理器周期性地更新网络延迟数据，但是这仍然不能反映网络上的实时变化。另外，该方法并没有考虑端点服务的负载情况。

图 7-17 描述了性能策略的工作原理。在性能策略方式下，流量管理器维护一个网络延迟表。流量管理器总是选择网络延迟最小的数据中心。

（1）当客户访问网站时，首先发送 DNS 请求解析域名 www.contoso.com。

（2）www.contoso.com 被解析为 contoso.trafficmanager.net。

（3）客户的 IP 地址属于地址段 2，流量管理器返回西欧数据中心网站给客户，因为该客户距离西欧数据中心最近。

（4）DNS 解析结果返回给客户。

（5）客户端收到 DNS 解析结果，HTTP 请求直接发往西欧数据中心网站，不经过流量管理器。

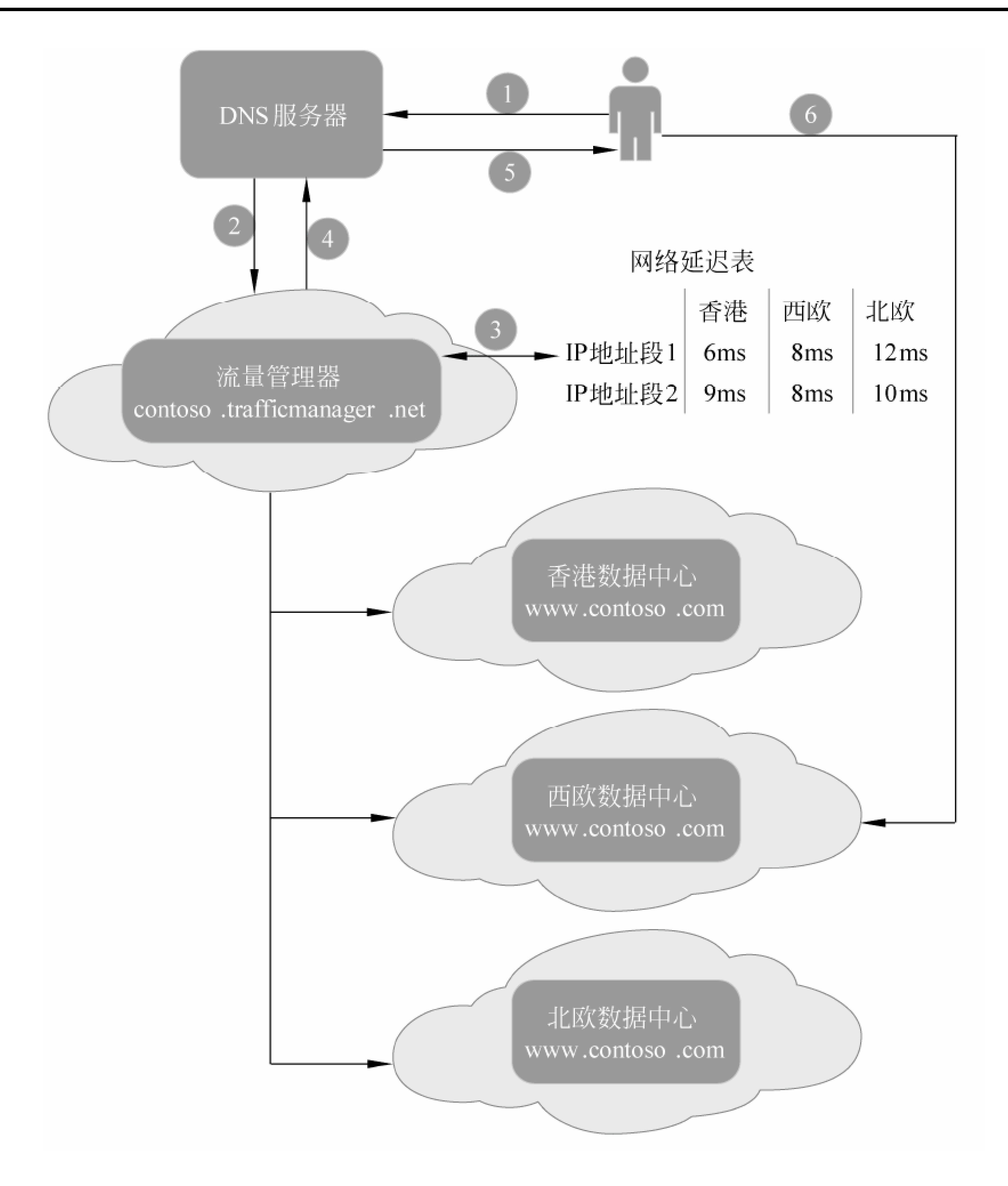


图 7-17　性能策略的工作原理

7.3.2.4　双重负载均衡

如果网站运行在多个实例上，Microsoft Azure 网站本身提供了在同一数据中心多个实例之间的故障转移和负载均衡功能。流量管理器为分布在不同数据中心的网站提供故障转移和负载均衡功能。图 7-18 详细描述了这种架构。

7.3.3　将流量管理器集成到 Azure 网站

下面用一个简单的例子来演示如何使用 Azure 流量管理器提高 Azure 网站的可用性和性能。

7.3.3.1　创建并配置流量管理器

1．创建流量管理器配置文件

（1）登录到 Microsoft Azure 管理门户网站。

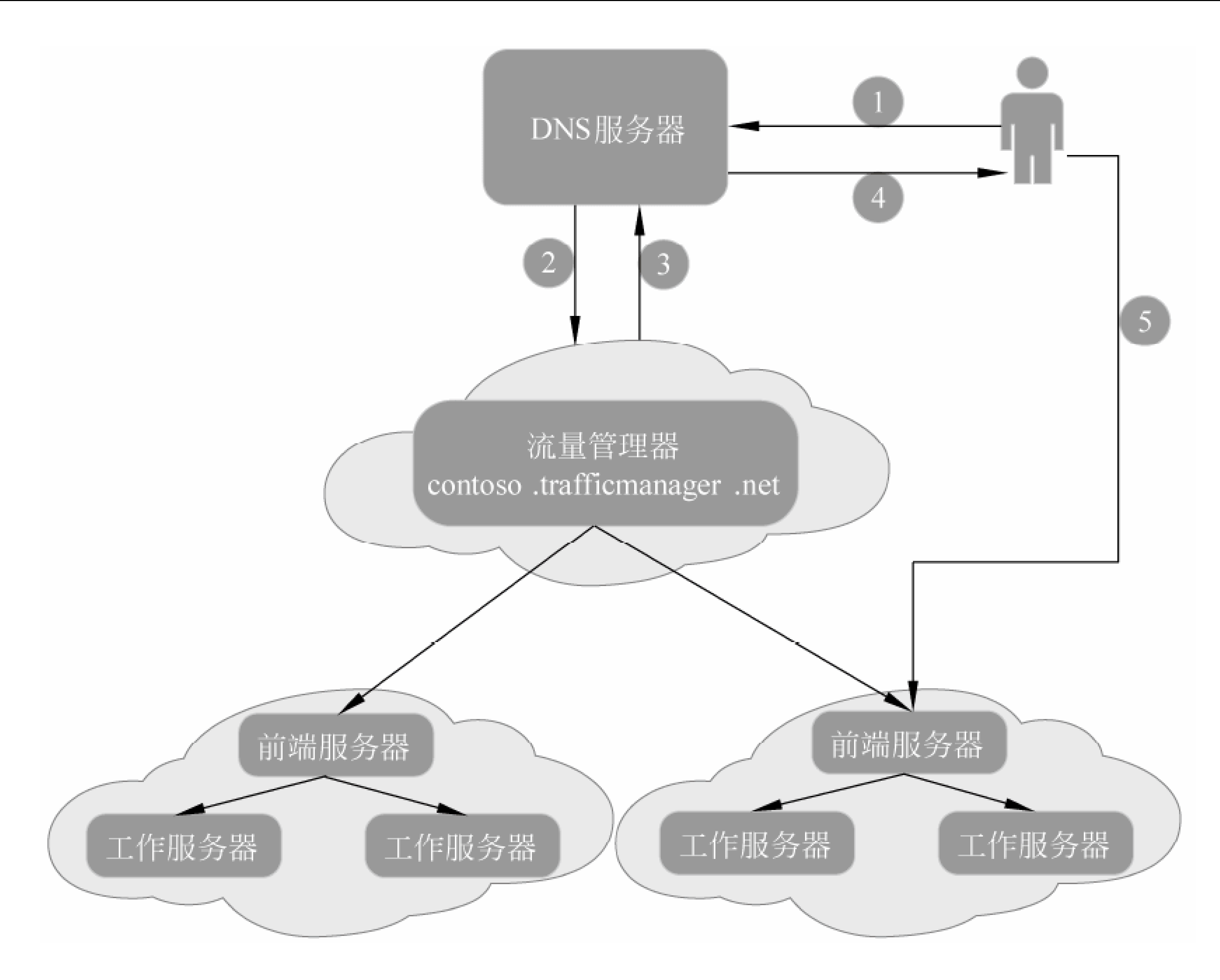


图 7-18　双重负载均衡

（2）如图 7-19 所示，单击页面左下角的创建按钮，选择 TRAFFIC MANAGER→"快速创建"，指定一个流量管理器的名称，这里 antarestest 为例。



图 7-19　创建流量管理器

2．配置自有域名

在 DNS 服务器中配置一个 CNAME 记录，将自有域名指向流量管理器域名：

```
www.websites.net.cn CNAME   antarestest.trafficmanager.net
```

3．创建 Azure 网站

可以根据业务需要在不同的数据中心创建网站。在下面的例子中，在香港数据中心创建了一个网站 waws-tm-hk，在美国东部数据中心创建一个网站 waws-tm-eu。

流量管理器只支持标准模式的网站，因此在进行下一步之前，需要将网站升级为标准

模式。关于如何将网站升级为标准模式，请参考第 2 章。

4．添加流量管理器终结点

（1）在“流量管理器配置”页面，单击上一步创建的流量管理器配置文件名称，打开配置页面。

（2）单击顶部的“终结点”，单击“添加终结点”，如图 7-20 所示，“服务类型”选择“网站”，然后选择要加入的网站名称。注意：

- 只支持标准模式的网站。
- 每个区域只能选择一个网站。

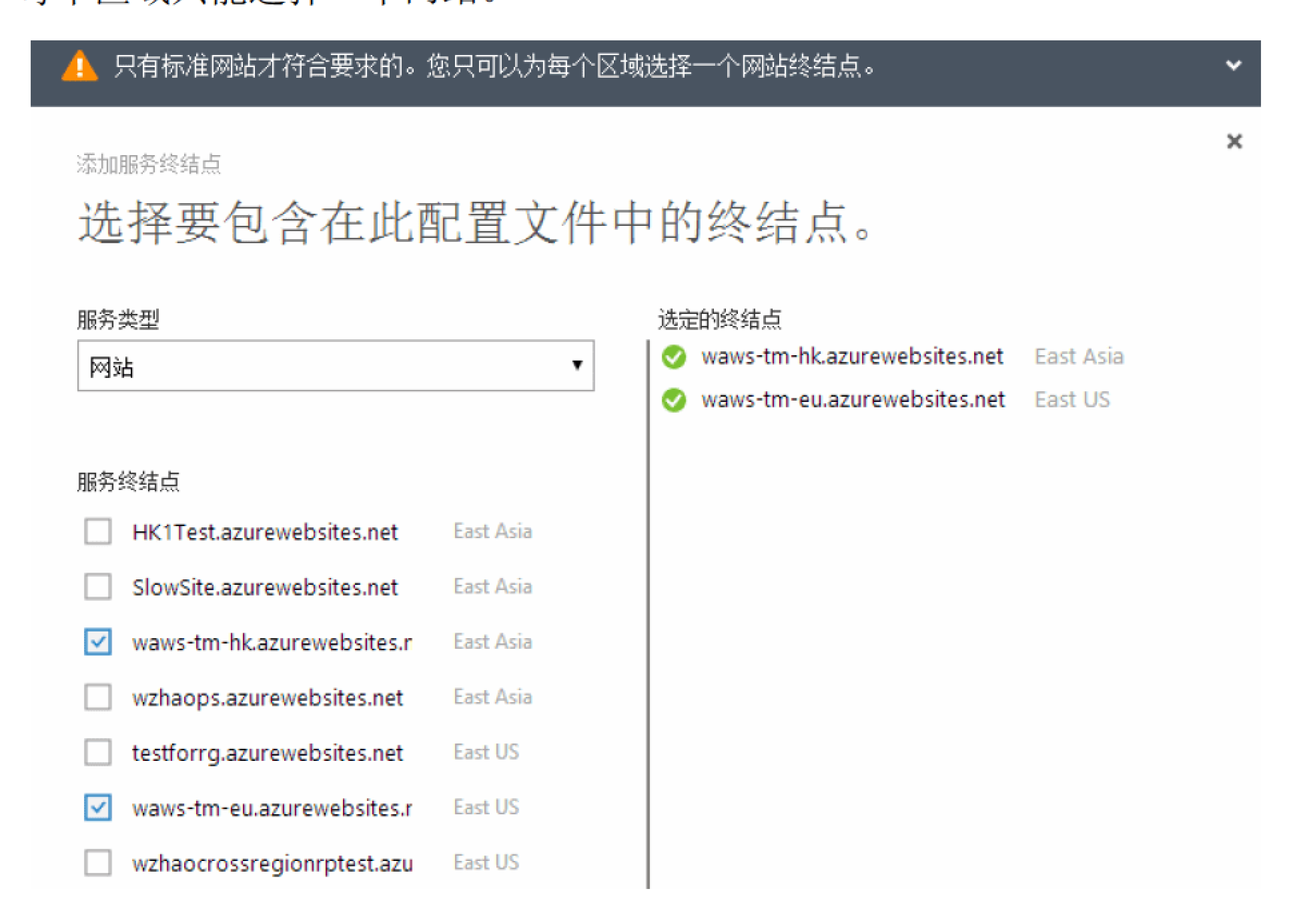


图 7-20 选择终结点

完成后，如图 7-21 所示，会看到两个网站终结点已经配置成功。



图 7-21 终结点状态

5．配置 Azure 网站自定义域名

如图 7-22 和图 7-23 所示，将自有域名绑定到已经创建的两个网站 waws-tm-hk 和

waws-tm-eu。由于前面已经将自有域名与流量管理器的域名相关联，因此，不需要额外的 DNS 配置。



图 7-22　香港数据中心网站配置自有域名



图 7-23　美国东部中心网站配置自有域名

注意：如果没有将自有域名绑定到网站，通过流量管理器访问时，Azure 网站的前端服务器将返回 404 错误。

7.3.3.2　创建并部署应用

（1）运行 Visual Studio 2013，选择“文件”→“新建”→“项目”。

（2）如图 7-24 所示，在新建项目窗口中选择 ASP.NET Web 应用程序，选择.NET Framework 4.5，命名为 UsingAzureTM，单击“确定”按钮。

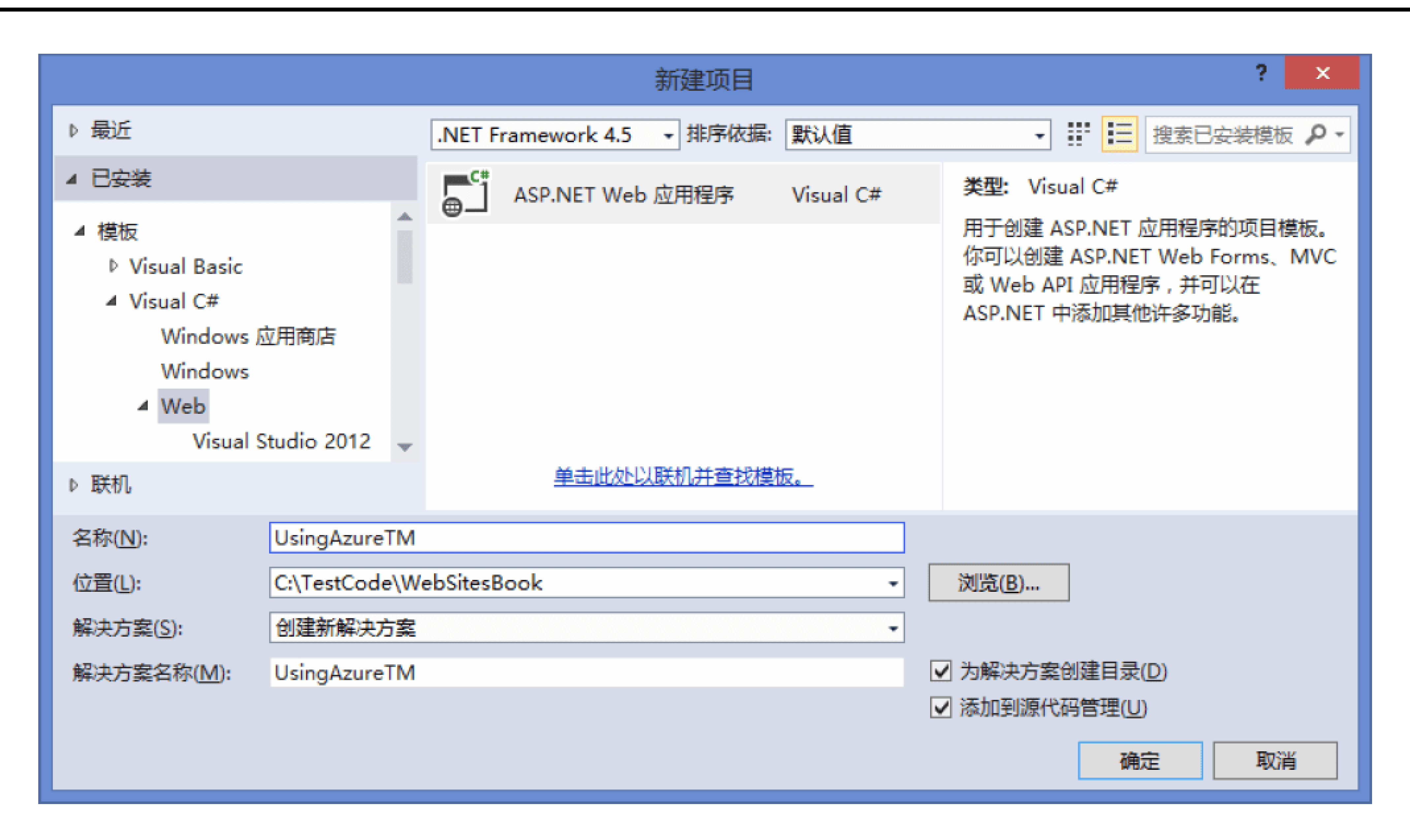


图 7-24 创建新的 ASP.NET 应用

（3）如图 7-25 所示，选择 Empty，单击“确定”按钮，创建项目。

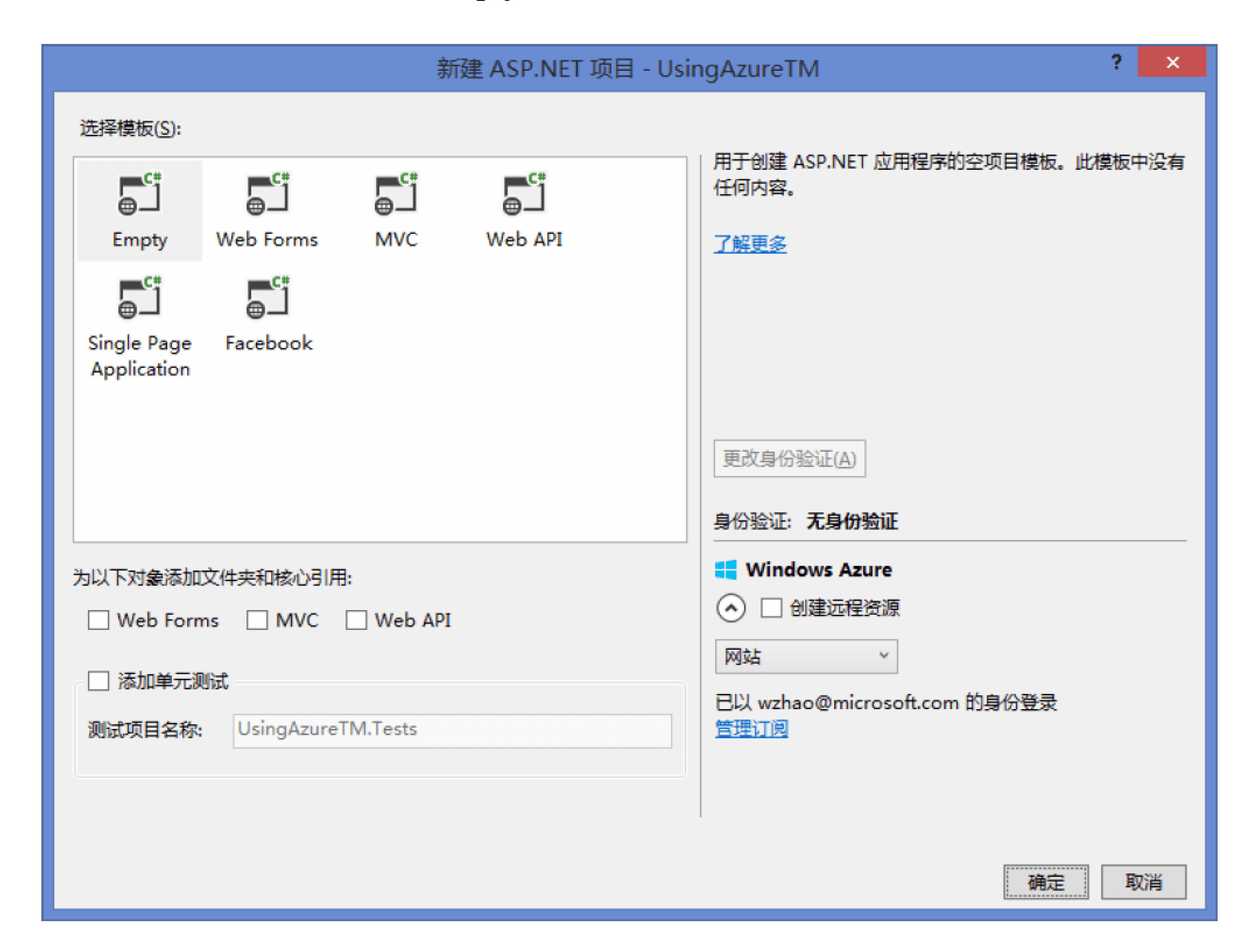


图 7-25 选择 Empty 模板

（4）在解决方案资源管理器中右击 UsingAzureTM 项目，选择“添加”→“新建项”，如图 7-26 所示，选择“Web 窗体”，将文件命名为 default.aspx。

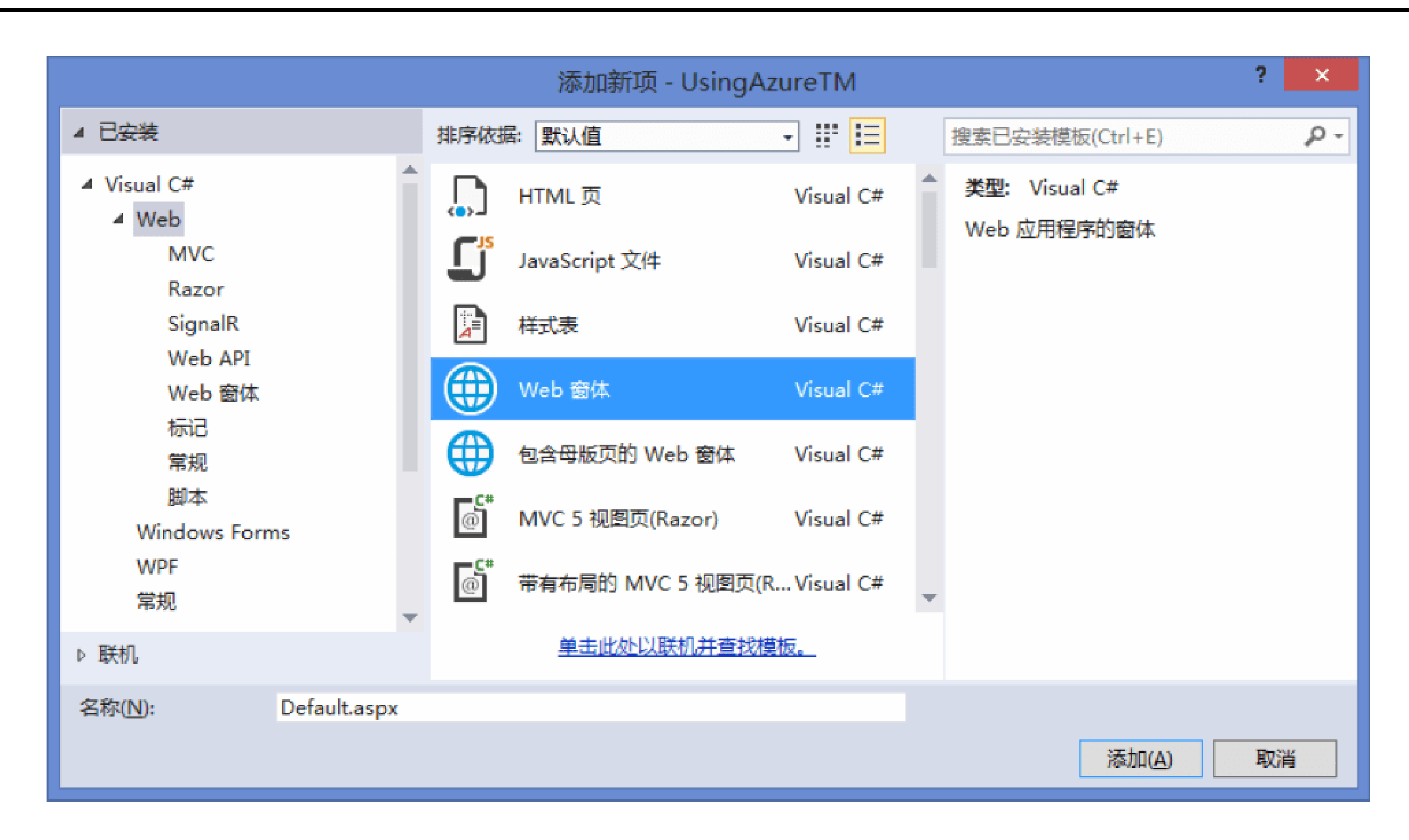


图 7-26　新建 Web 窗体

（5）在 Default.aspx.cx 中加入如下代码，代码显示网站的名称和客户端 IP 地址：

```
protected void Page_Load(object sender, EventArgs e)
{
    form1.InnerHtml = "<h2>Welcome to: "
     + Environment.GetEnvironmentVariable("APPSETTING_WEBSITE_SITE_NAME")
     + "</h2>";
    form1.InnerHtml += "<h2>You are from: "
     + Request.ServerVariables["REMOTE_ADDR"] + "</h2>";
    form1.InnerHtml += "<h2>Now is : "
        + System.DateTime.UtcNow.ToString() + "</h2>";
}
```

（6）将应用分别部署到之前创建的两个网站 waws-tm-hk 和 waws-tm-eu。关于如何从 Visual Studio 中部署应用到 Azure 网站，请参考第 4 章的内容。

（7）简单测试应用。打开 IE，浏览 http://waws-tm-hk.azurewebsites.net，网页显示 waws-tm-hk。浏览 http://waws-tm-eu.azurewebsites.net，网页显示 waws-tm-eu。

7.3.3.3　配置流量管理器的负载均衡方法

1．性能方法

流量管理器默认使用性能负载均衡方法。首先来测试性能负载均衡方法。在性能模式下，流量管理器默认将客户请求分发到与客户网络之间延迟最小的数据中心。本书作者使用家庭宽带访问 http://antarestest. trafficmanager.net，如图 7-27 所示，请求被分发到位于香港数据中心的 waws-tm-eu 站，因为家庭网络距离香港数据中心更近。



Welcome to: waws-tm-hk

You are from: 167.220.232.200

Now is : 6/16/2014 9:06:06 AM

图 7-27 请求被分发到香港数据中心

使用位于美国东部（夏洛特）的代理服务器时，如图 7-28 所示，请求被转发到 waws-tm-eu 网站。



图 7-28 请求被转发到美国东部数据中心

2. 循环方法

在循环（Round-Robin）模式下，客户请求被均匀分发到不同的数据中心。

首先，如图 7-29 所示，将负载平衡方法设置为循环法，并配置 DNS 生存时间（默认为 300s）。



图 7-29 负载均衡方法设置

打开 IE，访问 http://antarestest.trafficmanager.net，如图 7-30 所示，请求被分发到香港数据中心网站。

DNS 生存时间为 300s，因此需要等待 300s，使本地 DNS 的缓存失效。此时，再次访问 http://antarestest.trafficmanager.net，如图 7-31 所示，这次请求被分发到了美国东部数据中心站。



图 7-30　请求被转发到香港数据中心

图 7-31　请求被分发到美国东部数据中心

3. 故障转移

故障转移方式下，流量管理器维护一个网站的优先级列表。如图 7-32 所示，首先要在管理门户网站设置负载均衡方法和网站优先级列表。



图 7-32　设置网站优先级列表

配置完成后，访问 http://antarestest.trafficmanager.net，如图 7-33 所示，所有的请求都被分发到东亚数据中心。



图 7-33　请求被分发到香港数据中心

此时，将东亚数据中心的网站 waws-tm-hk 停掉，来模拟网站故障。此时回到流量管理器配置页面，可以看到网站状态已变为停止，如图 7-34 所示。



图 7-34　停止香港数据中心网站

此时，再次访问网站，如图 7-35 所示，所有用户请求都被分发至美国东部数据中心的网站 waws-tm-eu。



图 7-35　请求被分发到美国东部数据中心

注意：由于 DNS 缓存的生存时间为 300s（默认），在 DNS 缓存失效之前，还会连接到已经停止的 waws-tm-hk 网站。此时，访问网站时，会收到“服务不可用”错误。因此，即使使用了流量管理器的故障转移方式，当网站出现故障时，仍然可能有最多 300s 的时间无法访问网站。

另外，浏览器本身可能会缓存 DNS 的解析结果。需要查看浏览器文档来设置浏览器缓存 DNS 解析的时间。

7.4　集成内容传送网络

Microsoft Azure 内容传送网络（CDN）通过在遍布美国、欧洲、亚洲、澳大利亚和南美洲的众多物理节点上缓存静态内容，提供一个传送高带宽内容的全球性解决方案。有关 CDN 节点位置的完整列表请参阅下面的文档：

http://msdn.microsoft.com/en-us/library/azure/gg680302.aspx

通过 Microsoft Azure CDN，可以提供更好的性能和用户体验。如果网站的用户远离网站数据中心，网络延迟会降低用户体验，尤其是交互式的应用。通过 CDN 可以降低网络

延迟，提高响应速度。使用 CDN，网站在处理瞬时高负荷时会提供更快的响应（比如产品发布活动）。

对于 Azure 网站来讲，使用 CDN 除了可以提高性能和用户体验外，还可以扩大存储空间。标准模式的主机托管计划可以使用 50GB 的空间。但是，对于很多媒体类网站，50GB 的空间远远不够。

图 7-36 描述了一个 CDN 与网站集成的典型应用。静态内容，比如图片、视频和媒体文件等通过 CDN 访问，提高响应速度和客户体验；动态内容由网站直接服务。这样，既提高了静态文件的访问速度，同时减轻了网站服务器的访问压力。

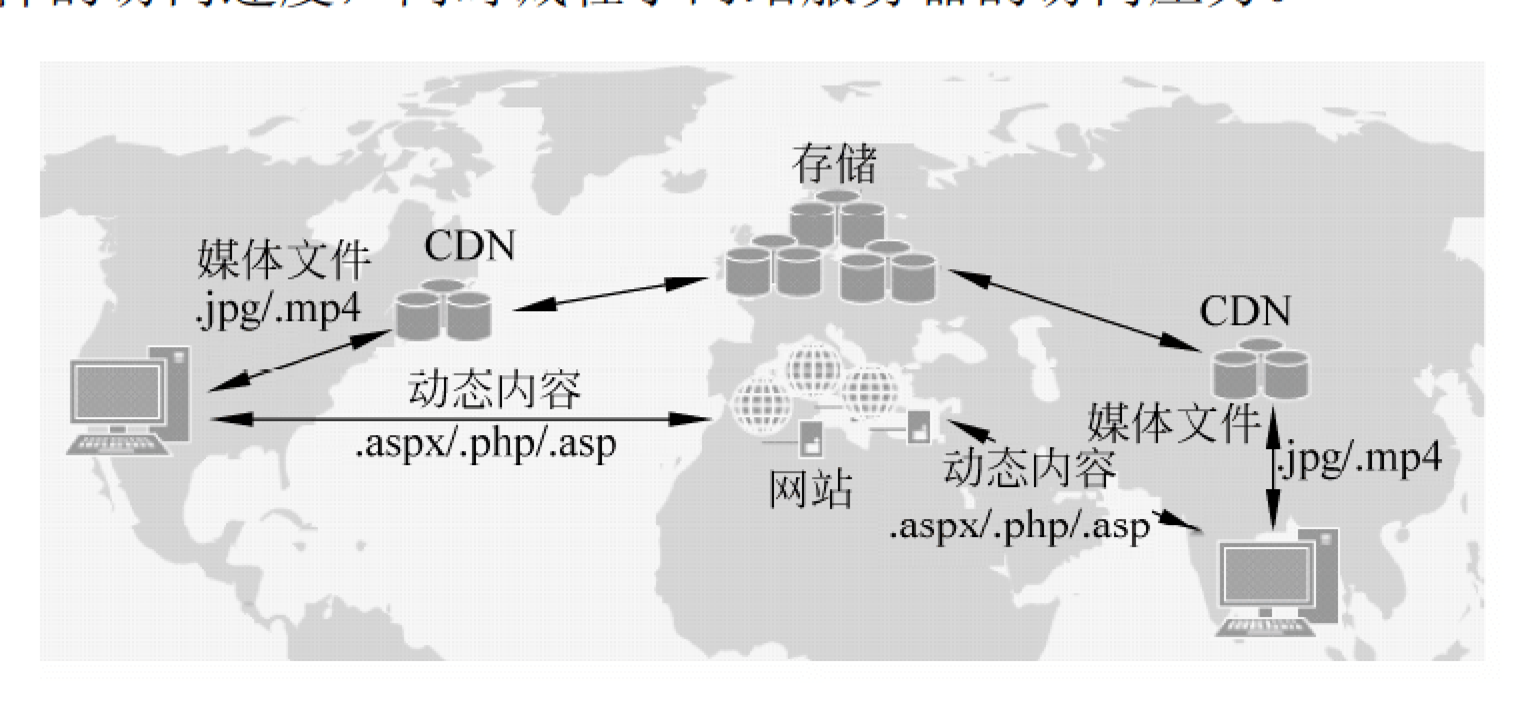


图 7-36　Azure 网站集成 CDN 架构

下面的例子演示了如何通过 URLRewrite 将 Azure 网站与 CDN 集成。通过 URLRewrite，无须修改和添加任何代码即可在 Azure 网站中集成 CDN 功能。URLRewrite 是 IIS 的一个模块，通过 URLRewrite，可以定义并执行重定向规则。关于 URLRewrite，请参考下面的文档：

http://www.iis.net/downloads/microsoft/url-rewrite

7.5　创建 Azure 存储账户

（1）登录到 Microsoft Azure 管理门户。

（2）在左下角单击“新建”，然后单击“存储”。

（3）单击“快速创建”。如图 7-37 所示，将显示“创建存储账户”对话框。



图 7-37　创建存储账户

（4）在 URL 字段中，输入一个子域名，域名可包含 3～24 个小写字母和数字，这里使用 antarescdn 作为域名。若要访问 Blob 中的资源，则要使用以下格式的 URL：

http://antarescdn.blob.core.chinacloudapi.cn/<mycontainer>

（5）从“位置/地缘组”下拉列表中为存储账户选择一个地理位置或者地缘组。

（6）单击“创建存储账户”。可能需要数分钟才能完成创建存储账户的过程。

现在，创建容器并上传文件。容器的名字必须与网站目录结构相同。比如，网站目录结构为

```
/wwwroot
    /images
        /archived
        /latest
    /videos
        /funny
```

那么，如图 7-38 所示，创建对应的容器。

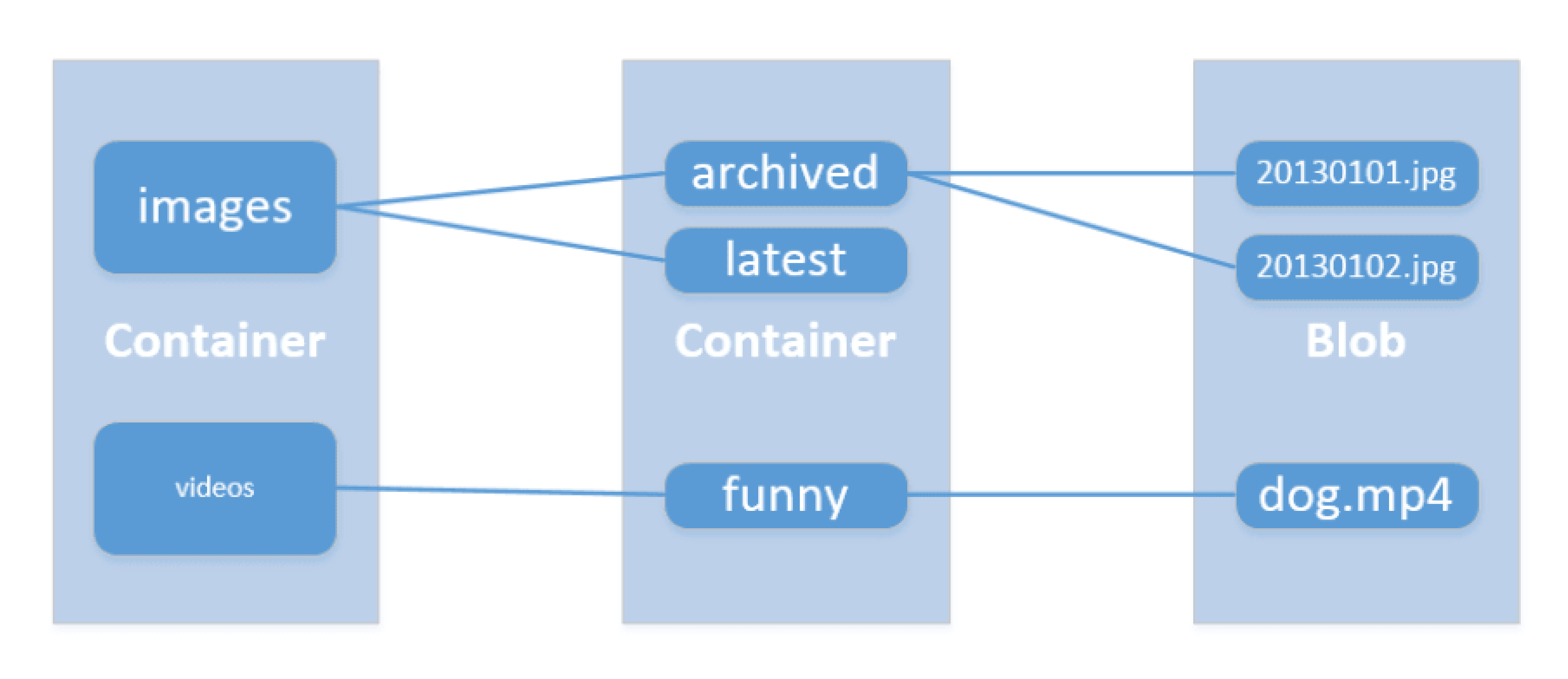


图 7-38　创建 Azure 存储容器

（1）登录 Microsoft Azure 管理门户网站。

（2）在左侧导航栏，单击 CDN，单击上一步创建的 antarescdn。

（3）在顶部导航栏，单击“容器”，单击“创建容器”。

（4）如图 7-38 所示，根据网站目录名称设置容器名称，访问权限设置为公共 Blob。

（5）重复第（3）步和第（4）步，创建其他容器。

（6）可以使用存储工具（比如 Azure Explorer）上传文件到 Azure 存储。可以访问下面的地址下载 Azure Explorer：

http://www.cerebrata.com/products/azure-explorer/introduction

（7）如果有很多文件需要上传，可以通过编写代码来自动上传文件。关于如何通过代码上传文件到 Azure 存储，请参考下面的文档：

http://azure.microsoft.com/en-us/documentation/articles/storage-dotnet-how-to-use-blobs/#upload-blob

（8）文件上传后，可以通过 URL 访问文件，比如：

https://antarescdn.blob.core.windows.net/images/logo.JPG

7.5.1　启用 CDN

一旦启用对存储账户的 CDN 访问，所有公开可用的对象会被 CDN 缓存。如果修改一个当前在 CDN 中缓存的对象，则用户会看到旧的内容。只有 CDN 缓存的内容失效后，才会更新对象的内容，此时才能通过 CDN 访问新内容。

（1）登录到 Microsoft Azure 管理门户网站。

（2）在左下角单击“新建”，然后单击“应用服务”，然后单击 CDN。

（3）单击“快速创建”。如图 7-39 所示，在“原始域”下拉列表中，选择上一步创建的 Azure 存储，单击“创建”。

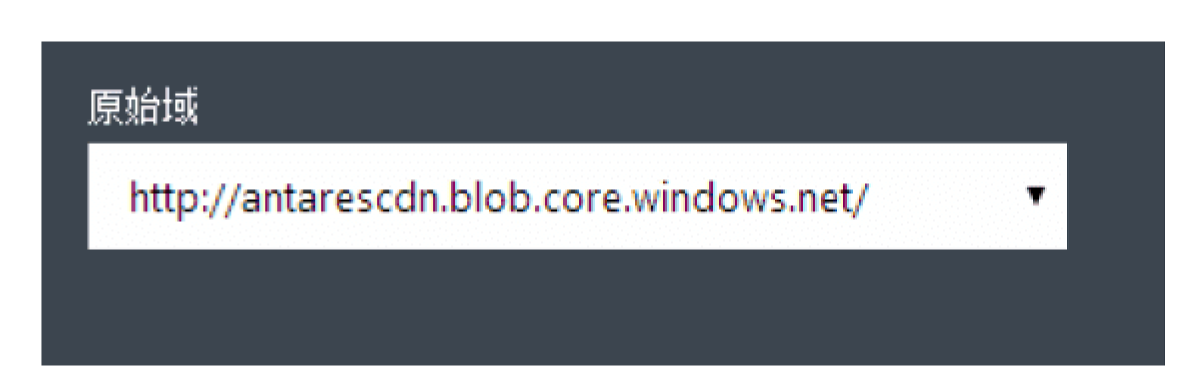


图 7-39　启用 CDN

（4）启用 CDN 后，可以通过下面的地址访问 Blob 中的文件。az610957 为自动生成的 CDN 的名称。

http://az610957.vo.msecnd.net/images/logo.JPG

7.5.2　创建 URLRewrite 规则

现在需要创建 URLRewrite 规则，将静态文件的访问重定向到 CDN。比如，客户通过 http://mysite.azurewebsites.net/images/logo.JPG 访问网站的图片文件，将请求重定向到 CDN 的 URL：http://az610957.vo.msecnd.net/images/logo.JPG。

下面的配置文件可以实现这个功能。需要将配置内容合并到网站的 web.config 文件。该规则将所有访问 images 和 videos 目录的请求都重定向到 CDN。需要将<*id*>替换为 CDN 名称。

```
<rewrite>
    <rules>
        <rule name="Images" stopProcessing="true">
            <match url="images/(.*)" />
            <action type="Redirect" url="http://<id>.vo.msecnd.net/{R:0}" />
        </rule>
        <rule name="Videos" stopProcessing="true">
            <match url="videos/(.*)" />
            <action type="Redirect" url="http://<id>.vo.msecnd.net/{R:0}" />
        </rule>
    </rules>
</rewrite>
```

如果希望更深入地了解 URLRewrite，请参考 URLRewrite 的文档：

http://www.iis.net/downloads/microsoft/url-rewrite

图 7-40 描述了启用 CDN 后的访问流程。亚洲的客户访问位于欧洲的网站时，网站返回 302 响应，将客户请求重定向到 CDN 网站，此时请求被 CDN 处理。通过该方案，既可以提高网站响应速度和客户体验，同时可以突破网站存储空间的限制。

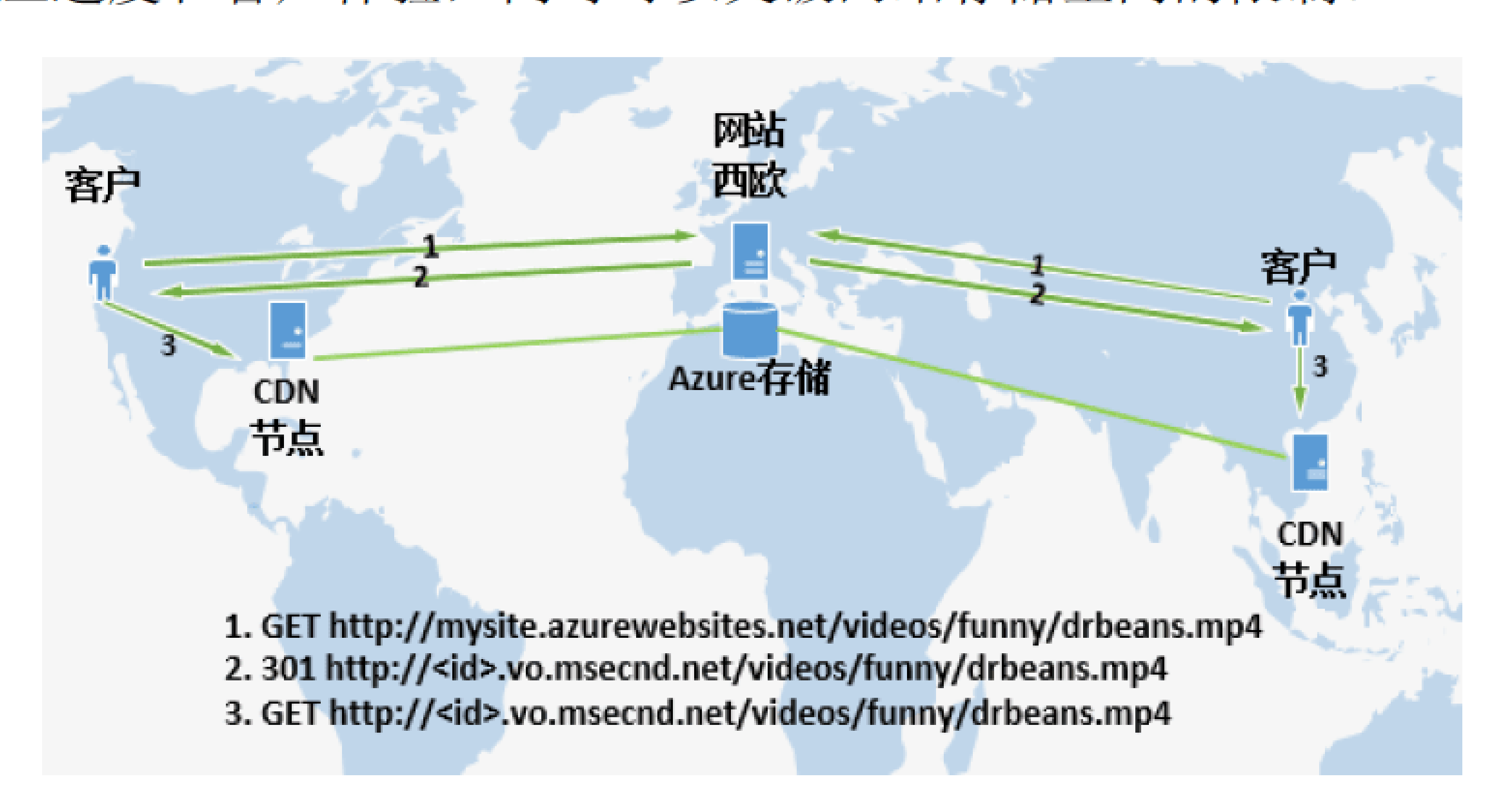


图 7-40　CDN 访问流程

7.5.3　集成 CDN 注意事项

CDN 是大小写敏感的产品。比如，有一个 abc.jpg 存放在 images 容器中，如果客户通过 http://az610957.vo.msecnd.net/images/Abc.jpg 来访问，那么 CDN 返回 404。最佳解决方案是 CDN 的所有容器中的文件都使用小写。然后，利用 URLRewrite 的规则将所有的 URL 都转换为小写。下面的 URLRewrite 规则使用了 tolower 方法将 URL 转换为小写。

```
<rewrite>
    <rules>
        <rule name="Images" stopProcessing="true">
            <match url="images/(.*)" />
            <action type="Redirect" url="http://<id>.vo.msecnd.net/{tolower:
            {R:0}}" />
        </rule>
        <rule name="Videos" stopProcessing="true">
            <match url="videos/(.*)" />
            <action type="Redirect" url="http://<id>.vo.msecnd.net/{tolower:
            {R:0}}" />
        </rule>
    </rules>
</rewrite>
```

该规则配置后，假如客户通过 http://mysite.azurewebsites.net/images/logo.JPG 访问网站的图片文件，我们将 HTTP URL 中的大写字母转换为小写字母，转换后重定向到 CDN 的 URL 只包含小写字母：http://az610957.vo.msecnd.net/images/logo.jpg。

关于更多的 CDN 最佳实践，请参考下面的文章：

http://blogs.msdn.com/b/windowsazure/archive/2011/03/18/best-practices-for-the-windows-azure-content-delivery-network.aspx

7.6　利用 Microsoft Azure 活动目录实现身份认证

Windows 活动目录是 Windows 系统的核心组件，其中一项主要的功能就是用户服务。它提供了管理用户域账户、用户信息、用户组管理、用户身份认证、用户授权管理等。通过活动目录登录到网络上的用户既能够获得身份验证，也可以获得访问系统资源所需的权限。企业内部应用通常使用 Windows 活动目录来实现 Web 应用的单点登录和访问控制。

在 Microsoft Azure 中，可以创建、运行并管理活动目录，并且可以与本地部署的活动目录进行同步，即可以自动同步您的本地用户到 Microsoft Azure 活动目录。

部署在 Microsoft Azure 中的应用可以利用 Microsoft Azure 活动目录来进行身份验证。在本节中，通过一个实例来演示如何使用 Visual Studio 2013 和.NET Framework 4.5 中的 Windows Identity Foundation（WIF）类开发一个 ASP.NET 应用。它将显示如何在 Azure 网站中集成 Microsoft Azure 活动目录来实现身份认证。

7.6.1　解决方案体系结构

图 7-41 描述了利用 Microsoft Azure 活动目录实现身份认证的认证流程。当客户访问基于 Azure 活动目录进行认证的网站时（步骤①），客户首先被重定向到登录页面（步骤②）。在登录页面，客户提供基于 Azure 活动目录的身份信息（用户名和密码）。身份信息被发送

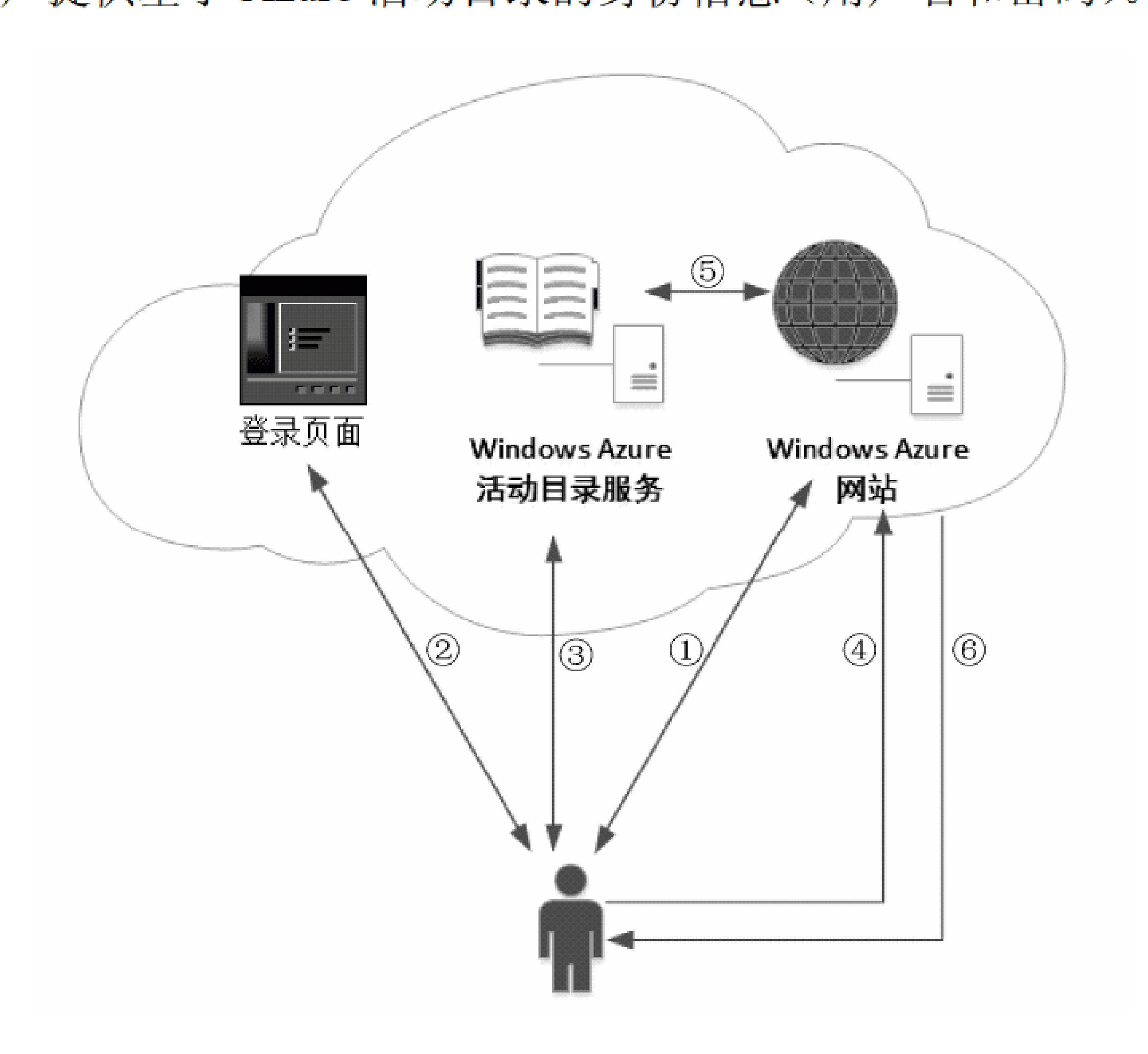


图 7-41　Azure 活动目录认证流程

到 Azure 活动目录进行认证。认证通过后，活动目录返回认证信息（步骤③），并再次将客户重定向到网站页面（步骤④）。此时，HTTP 请求中包含了认证信息。Azure 网站通过 Azure 活动目录验证客户的认证信息（步骤⑤）。验证通过后，处理请求，并返回 HTTP 响应（步骤⑥）。

下面，通过具体步骤演示如何在 Azure 网站应用中集成 Microsoft Azure 活动目录来实现身份认证。

7.6.2 创建 Azure 活动目录

7.6.2.1 添加目录

（1）登录到 Microsoft Azure 管理门户网站。

（2）点击左下角的“新建”按钮。

（3）选择“应用服务”→Active Directory→“目录”→“自定义创建”。

（4）如图 7-42 所示，提供一个名称和域名，并指定国家或地区。



图 7-42 添加 Azure 活动目录

（5）单击“完成”创建活动目录。

7.6.2.2 添加用户

（1）登录到 Microsoft Azure 管理门户网站。

（2）在左侧导航栏，单击 Active Directory。

（3）选择上一步创建的活动目录。

（4）单击顶部的“用户”，打开“用户”页面。

（5）单击底部命令栏的“添加用户”。

（6）如图 7-43 所示，提供用户名。

（7）单击“下一步”，如图 7-44 所示，提供客户的具体信息。修改“角色”选项，可以创建普通用户或者管理员用户。Azure 订阅用户默认是全局管理员用户。



图 7-43　指定用户名称和用户类型

图 7-44　指定用户具体信息

（8）单击“下一步”，然后单击“创建”。

（9）如图 7-45 所示，用户创建后，会自动生成一个临时密码。当用户第一次登录时，会要求用户重新设置密码。可以通过邮件的方式将临时密码发送给用户。



图 7-45　临时密码

（10）单击“完成”。

（11）如果需要添加更多用户，请重复上面的步骤。

如果需要批量创建用户，建议通过 Microsoft Azure PowerShell 来添加用户。具体信息请参考下面的文档：

Manage Azure AD using Windows PowerShell

http://technet.microsoft.com/en-us/library/dn194096.aspx

7.6.3 创建一个使用 Azure 活动目录认证的 ASP.NET 网站

下面演示如何创建一个 ASP.NET 网站。该网站将利用 7.6.2 节创建的 Azure 活动目录进行身份验证。

7.6.3.1 创建一个使用 Azure 活动目录认证的 ASP.NET 项目

（1）运行 Visual Studio 2013。

（2）依次选择菜单“文件”→“新建”→“项目”命令。

（3）在图 7-46 所示的“新建项目”对话框中，选择创建一个 Visual C# Web 应用（.NET Framework 4.5），单击“确定”按钮。

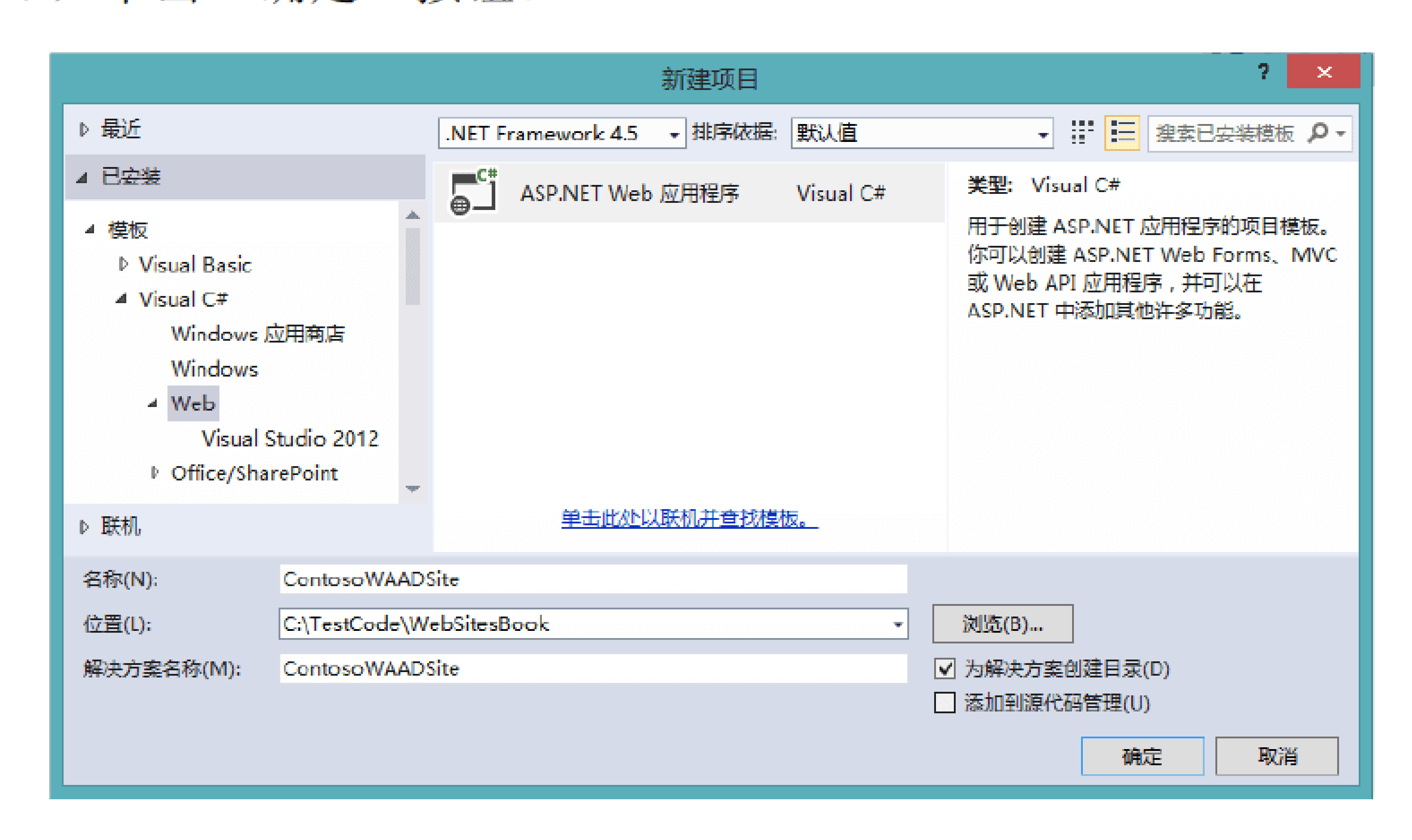


图 7-46 创建新的 ASP.NET 应用

（4）在“新建 ASP.NET 项目”窗口，选择 Web Forms 模板。

（5）在窗口右侧，选择“更改身份验证”，如图 7-47 所示，在“更改身份验证”窗口，选择“组织账户”。在本例中，选择“云-单个组织”。在“域”一栏中，填入上一步创建的 Azure 活动目录。在本例中，活动目录是 ContosoWAAD.onmicrosoft.com。“访问级别”设置为“单一登录”。



图 7-47 选择认证方式

（6）单击“确定”按钮。此时，会弹出“登录到 Azure Active Directory”的认证窗口，选择“使用另外一个账户”，输入全局管理员账户的用户名和密码。

注意：如果是初次登录，首先会被引导到修改密码页面。

（7）认证成功后，应用与 Azure 活动目录绑定成功，如图 7-48 所示。

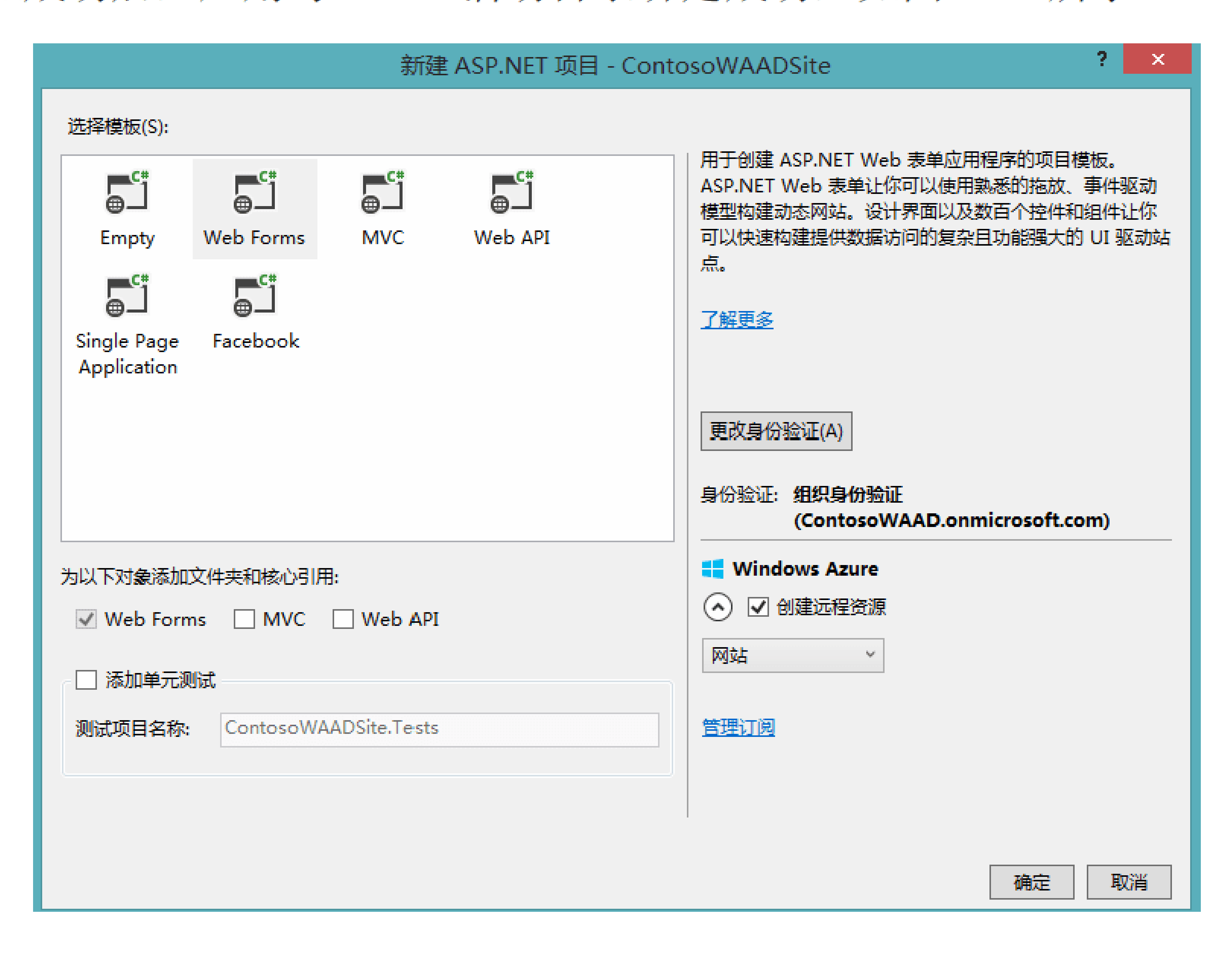


图 7-48　集成 Azure 活动目录认证的项目

（8）在 Microsoft Azure 区域，选择“创建远程资源”，资源类型设置为“网站”。单击“确定”按钮。

（9）如图 7-49 所示，在“配置 Microsoft Azure 站点”窗口，单击“登录”按钮，使用 Azure 订阅登录。

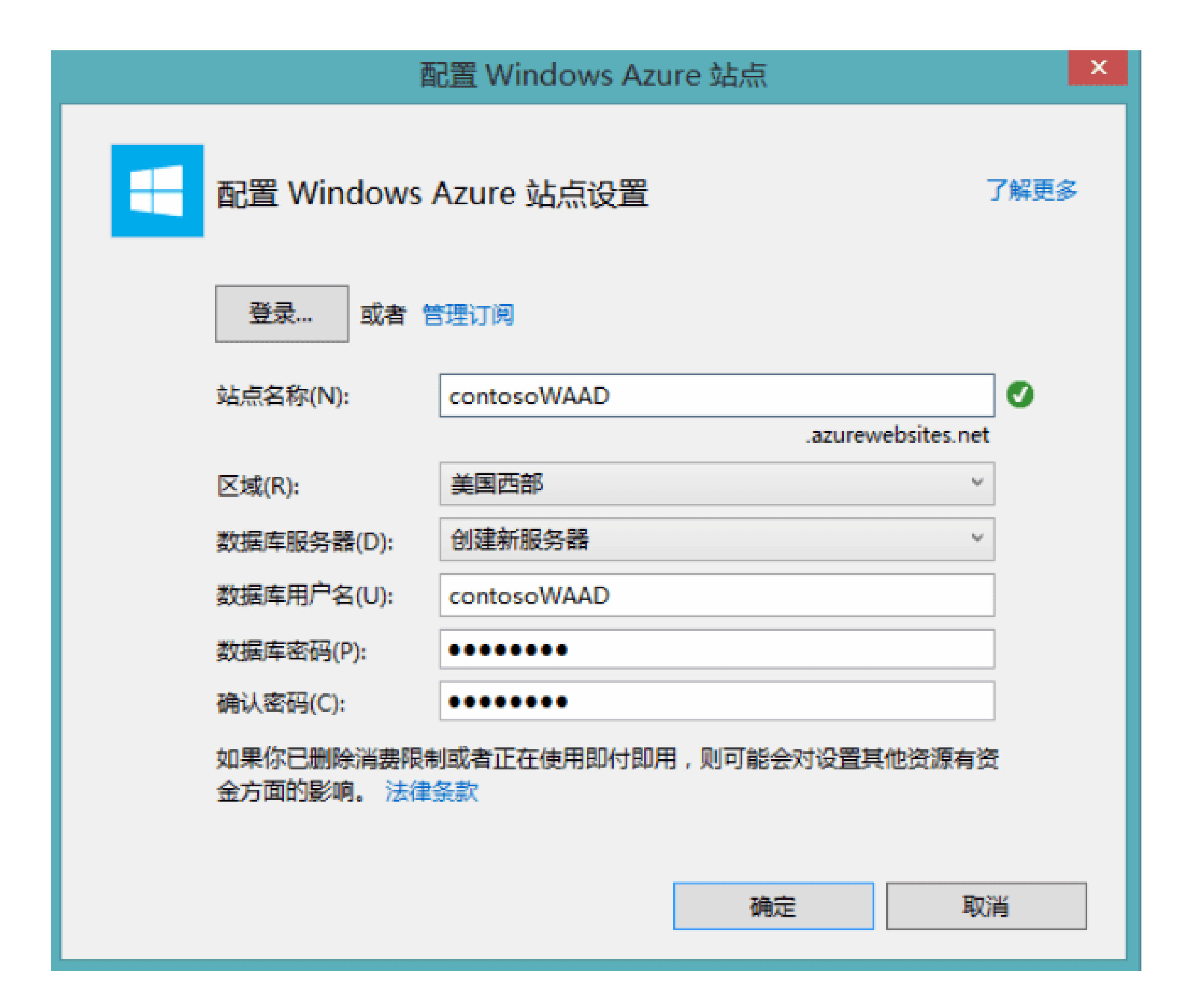


图 7-49　配置 Azure 站点

指定网站的数据中心和数据库服务器的选项。可以选择利用已有数据库服务器，或者创建新的数据库服务器。在这里，选择“创建新服务器”。

（10）单击“确定”按钮创建项目和 Azure 网站。

7.6.3.2　在本地环境测试网站

（1）在 Visual Studio 中，按下 Ctrl+F5 键启动 IE，在本地浏览网站。

（2）如果看到证书错误，请忽略。选择继续浏览。

（3）现在被重定向到登录页面。

（4）使用在第一步创建的 Azure 活动目录账户登录。

注意：

① 如果第一次使用该账户登录，会被强制要求更改密码。

② 如果忘记了临时密码，可以登录 Azure 网站重新生成临时密码。

（5）登录后，会看到账户名称显示在网站右上方，如图 7-50 所示。

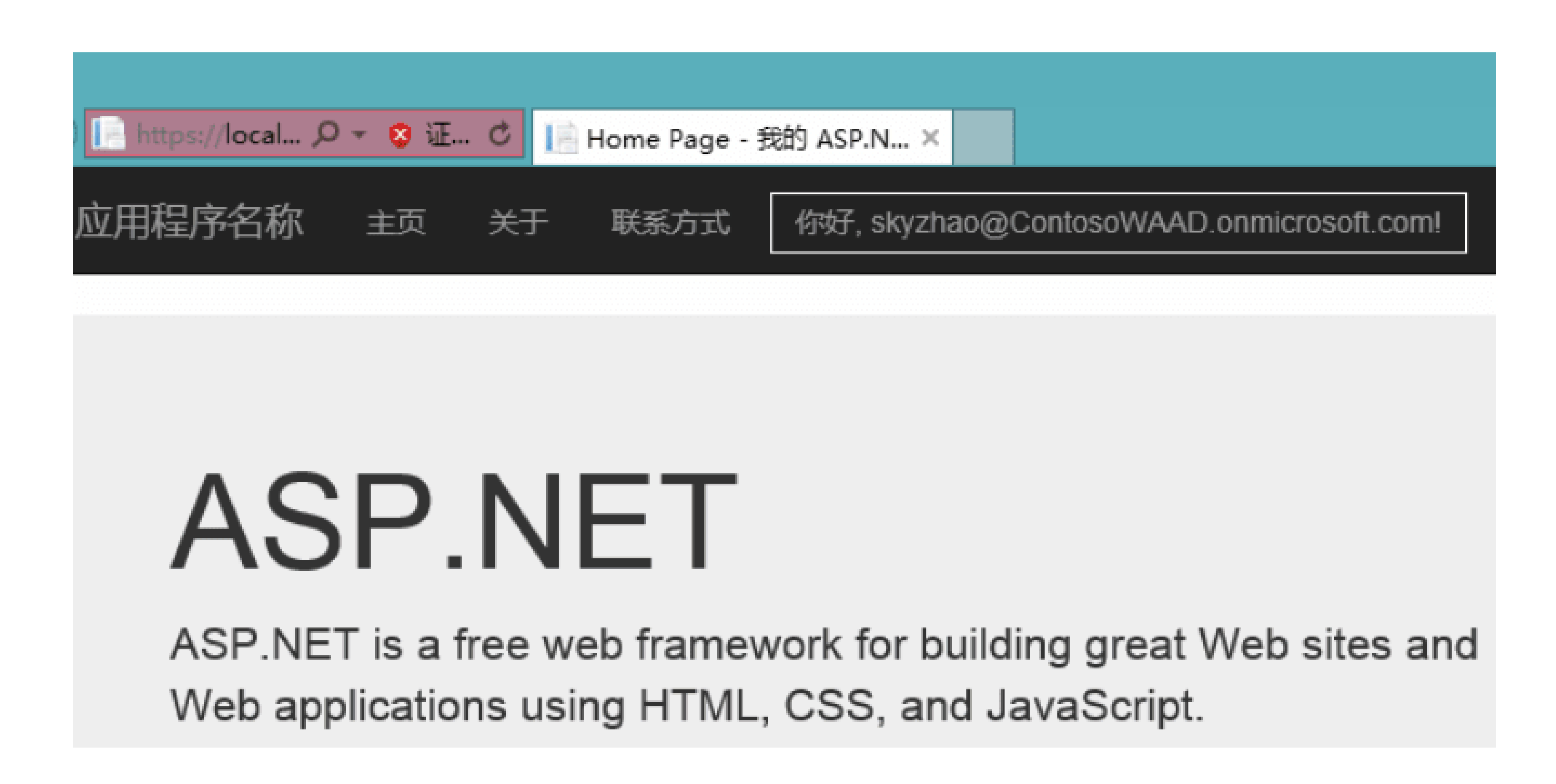


图 7-50　本地测试 Azure 活动目录网站

7.6.3.3　将项目部署到 Azure 网站

（1）在 Visual Studio 中选择菜单“生成”→“发布”命令。

（2）在“连接”窗口，单击“验证连接”，确保可以连接到 Azure 网站发布服务。

（3）单击“下一步”按钮。

（4）在“设置”窗口，如图 7-51 所示，指定替换数据库连接字符串。

（5）单击“发布”，将项目发布到 Azure 网站。

（6）项目发布成功后，Visual Studio 自动打开 IE 并浏览网站。此时 IE 显示网站错误，这是因为 Azure 活动目录没有与 Azure 网站连接。请忽略该错误。下面将通过修改 Azure 活动目录配置来修复该错误。



图 7-51　发布 Web 项目到 Azure 网站

7.6.4　连接 Azure 网站与 Azure 活动目录

下面，将配置 Azure 活动目录的选项以便 Azure 网站能够使用 Azure 活动目录进行身份认证。

（1）登录到 Azure 管理门户网站。

（2）在左侧导航栏选择 Active Directory。

（3）单击之前创建的活动目录，在本例中是 ContosoWAAD。

（4）在顶部导航栏，单击“应用程序”。

（5）如图 7-52 所示，会看到上一步创建的应用。此时，应用程序 URL 执行 localhost。



图 7-52　应用程序 URL

（6）单击 ContosoWAADSite 应用，打开“应用程序配置”页面。

（7）在顶部导航栏，单击“配置”。

（8）如图 7-53 所示，在页面顶部，修改“登录 URL”，使其指向网站。在本例中，是 https://contosoWAADSite.azurewebsites.net/。

注意：必须是 https。

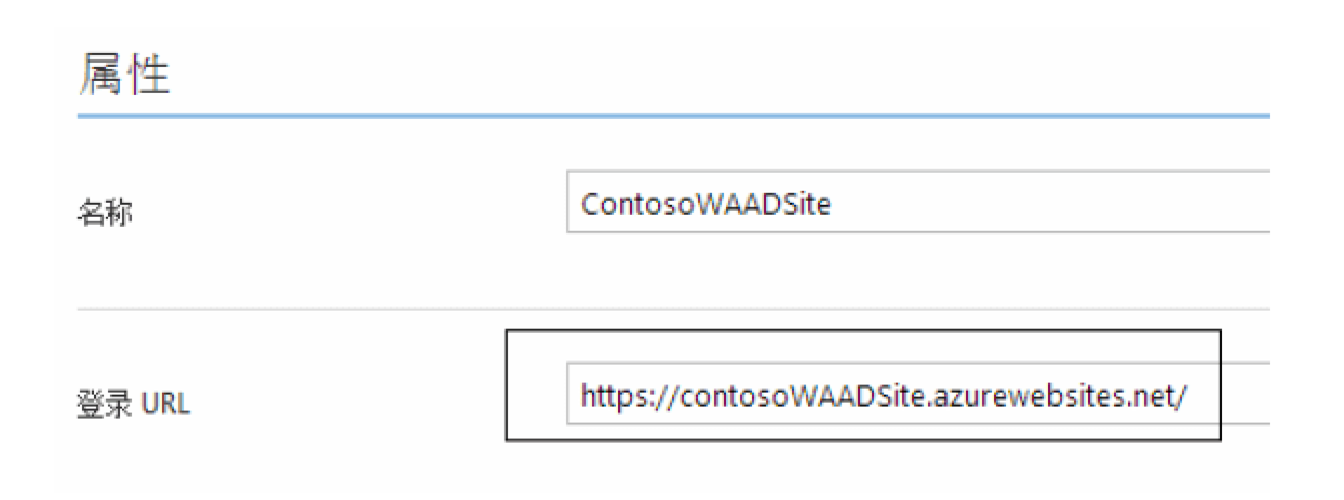


图 7-53 指定登录 URL

（9）向下滚动页面，在“单点登录”区域，修改“回复 URL”。同样，回复 URL 也需要指向网站，在本例中，是 https://contosoWAADSite.azurewebsites.net/，如图 7-54 所示。



图 7-54 指定回复 URL

注意：必须是 https。

（10）单击页面底部命令栏的“保存”。

7.6.5 测试 Azure 网站

现在，所有的配置工作已经完成，Azure 网站已经与 Azure 活动目录集成。此时，访问 Azure 网站必须提供有效的 Azure 活动目录账户。

（1）打开 IE，浏览网站。本例中，网站地址是 https://contosoWAADSite.azurewebsites.net/

（2）现在，重定向到登录页面。

（3）使用之前创建的 Azure 活动目录账户登录。

注意：

① 如果第一次使用该账户登录，会被强制要求更改密码。

② 如果忘记了临时密码，可以登录 Azure 网站重新生成临时密码。

（4）登录后，在页面右侧，可以看到账号信息，如图 7-55 所示。



图 7-55 采用 Azure 活动目录认证的网站

7.6.5.1　注意事项

（1）使用 Azure 活动目录认证的站点，只能通过 HTTPS 方式访问。

（2）如果遇到登录问题，可能是当前浏览器登录了其他网站。可以采用 InPrivate 浏览方式。如图 7-56 所示，右击 IE，选择“开始 InPrivate 浏览”。

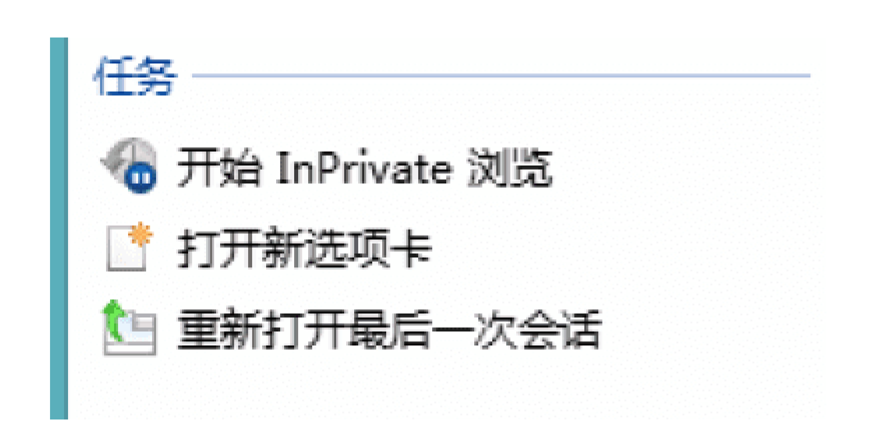


图 7-56　采用 InPrivate 浏览

7.7　通过混合连接访问企业内部资源

云计算的敏捷性和经济高效性越来越有吸引力，借助云计算，企业能够对客户需求和市场竞争更快地做出反应。因此，越来越多的企业选择将应用迁移到云计算平台。同时，毫无疑问，很多时候企业希望内部 IT 系统和云计算平台的系统能够无缝集成，互联互通。比如：

- 企业选择将新的应用部署到云计算平台。新的应用需要访问企业内部 IT 系统的资源，比如企业市场数据库。
- 在企业中，某些特定的应用基于法规、安全性、性能和可用性等考虑，不能迁移到云计算平台。这些应用需要与部署到云计算平台的应用互通互联。
- 另外，某些大型系统的迁移是一个长期的过程。企业根据需要将大型系统的功能逐步地分模块迁移到云计算平台。在迁移过程中，已经迁移到云计算平台的模块和仍然位于企业内部数据中心的各个模块之间需要互联互通。

7.7.1　Azure 混合连接

Microsoft Azure 服务总线能够将部署在 Azure 的应用与企业内部应用甚至是合作伙伴的系统无缝集成。比如，可以从部署在 Microsoft Azure 的应用访问部署在企业内部的商业应用，如 Microsoft CRM 系统、SQL 数据库等。图 7-57 描述了这种架构。

下面的文档详细介绍了如何通过服务总线将企业内部应用和部署在 Azure 的应用集成起来：

http://www.windowsazure.cn/zh-cn/develop/net/tutorials/hybrid-solution/

Microsoft Azure 服务总线提供了一整套框架模块。需要根据业务需要，开发基于 Microsoft Azure 服务总线的应用将部署在 Microsoft Azure 的应用与企业内部应用无缝集成。开发者需要熟悉 Azure SDK、服务总线和 WCF 等。对于某些中小企业或者需要一个

暂时过渡方案的企业来讲，服务总线开发周期长，投入比较大，并不是一个最佳选择。

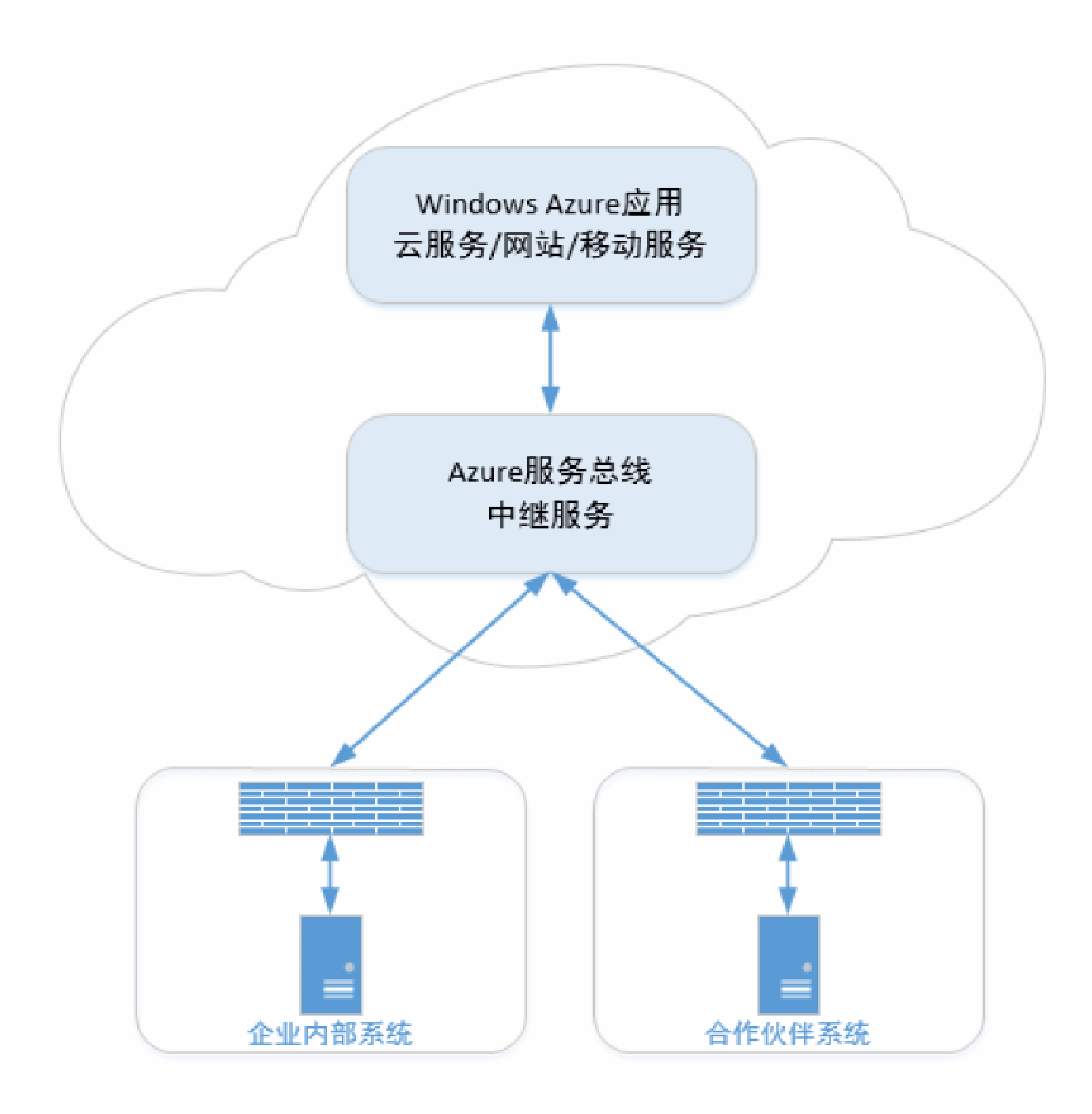


图 7-57　基于服务总线实现 Azure 混合云解决方案

Microsoft Azure BizTalk 服务提供了混合连接（hybrid connection）解决方案，完美地解决了该问题。混合连接方案基于 Azure 服务总线，对 Azure 服务总线进行了封装。使用混合连接方案，不再需要额外的开发工作，大大提高了部署速度，并降低了成本。

BizTalk 混合连接具有如下优点：

- Azure 网站和移动服务可以安全地访问现有的部署在企业内部的数据和服务。
- 多个 Web 站点或移动服务可以共享一个混合连接来访问内部部署资源。
- 无须更改企业网络，如配置一个 VPN 网关，或开发额外的防火墙端口。
- 使用混合连接的应用程序只能访问混合连接指定的企业内部资源。
- 可连接到任何一个使用静态 TCP 端口的部署在企业内部的服务，如 SQL Server、MySQL、基于 HTTP 的 Web API 以及 Web Service。
- 可用于与 Azure 网站支持的所有框架（.NET、PHP、Java、Python、Node.js）和 Azure 移动服务（Node.js、.NET）。
- 通过混合连接，网站和移动服务访问本地资源的方式与企业内部部署的其他应用完全相同，比如企业内部部署的应用与部署在 Azure 网站的应用使用相同的连接字符串访问相同的数据库。

图 7-58 描述了部署在 Azure 网站上的应用访问企业内部数据库服务器的架构。在运行 Azure 网站的虚拟机上运行一个混合连接代理应用，在企业内部数据中心的数据库服务器上运行混合连接管理器。

7.7.2　应用实例架构

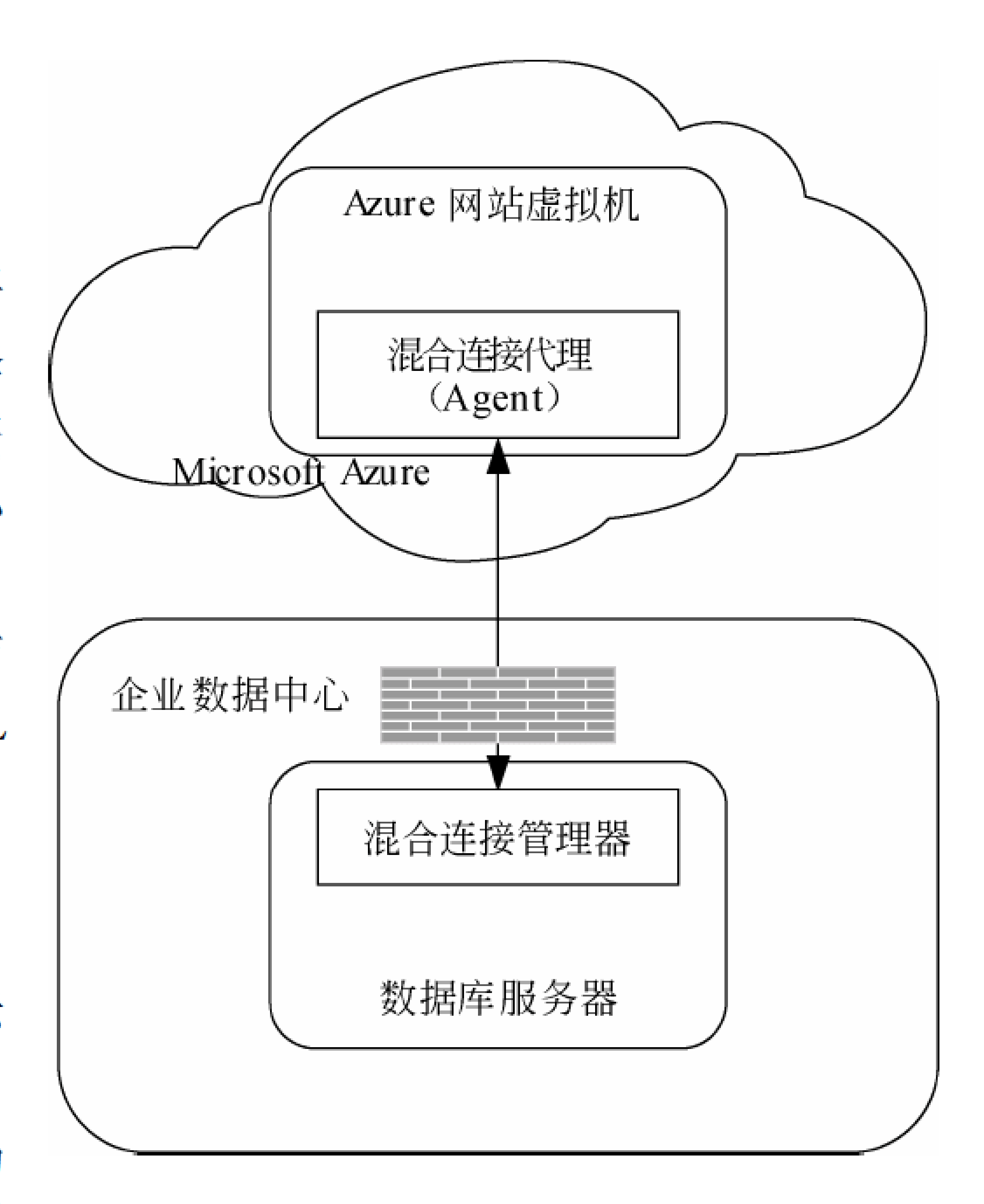


图 7-58　混合连接架构

通过混合连接，企业无须部署 VPN，部署在 Azure 网站中的应用无须配置防火墙以及修改代码即可轻松、安全访问位于企业内部的资源。通过混合连接集成网站与企业内部数据中心的主要步骤如下：

（1）创建混合连接，指定允许 Azure 网站访问的企业内部资源，比如 Web Service 或者 SQL Server。

（2）将 Azure 网站链接到混合连接。

（3）在本地服务器上安装混合连接管理器。

下面以一个企业产品 API 系统为例，演示如何通过混合连接将网站与企业内部系统集成。该产品 API 系统为很多企业内部系统比如市场部、生产部门、财务部门等提供服务。企业计划逐步将所有应用都迁移到 Azure 网站。在第一阶段需要将市场部门应用迁移到 Azure 网站。具体架构如图 7-59 所示。

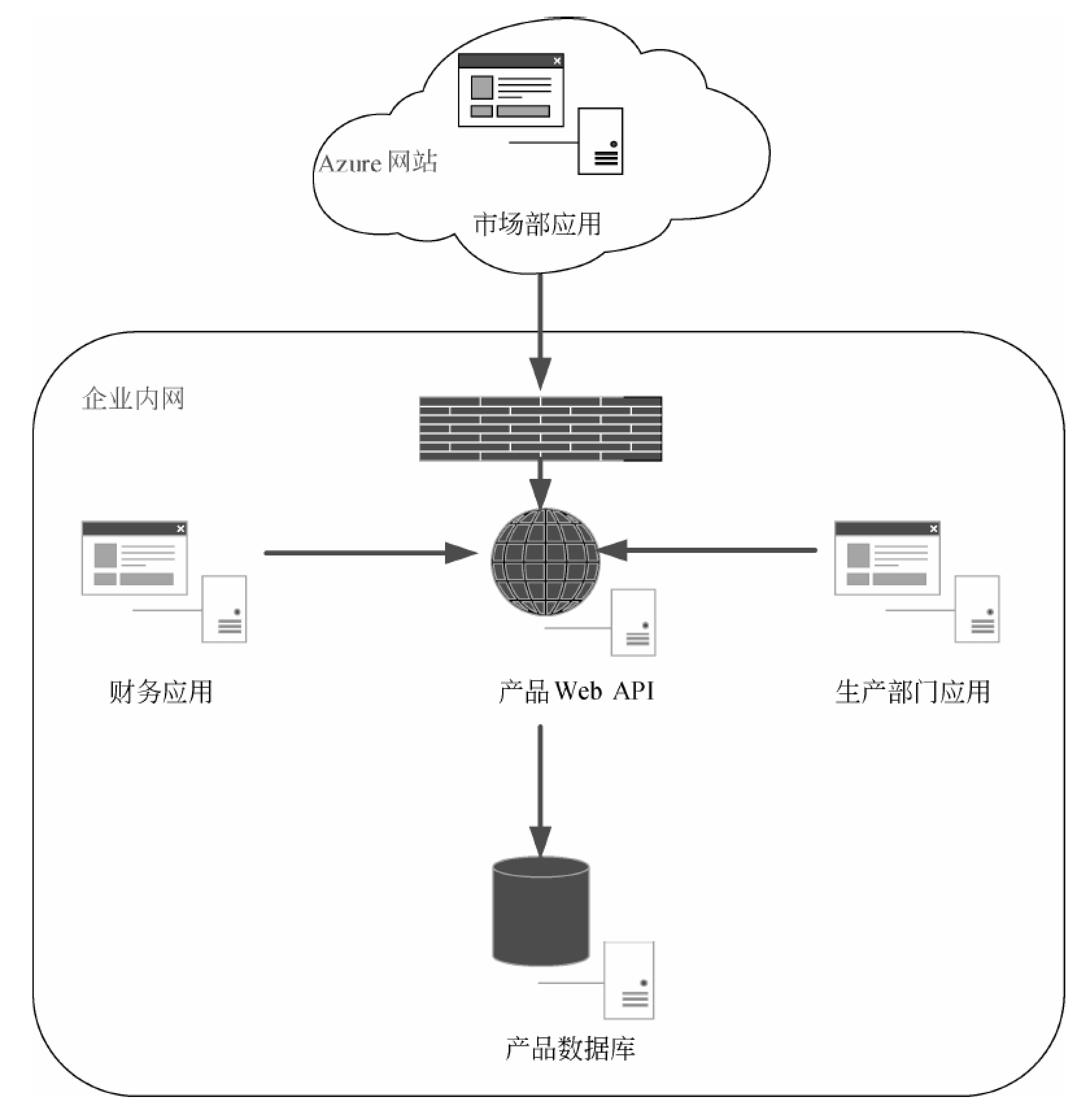


图 7-59　产品 API 系统架构

产品 API 服务通过 http://ProductAPI/访问，比如 http://ProductAPI/api/products 返回所有产品及价格，如图 7-60 所示。

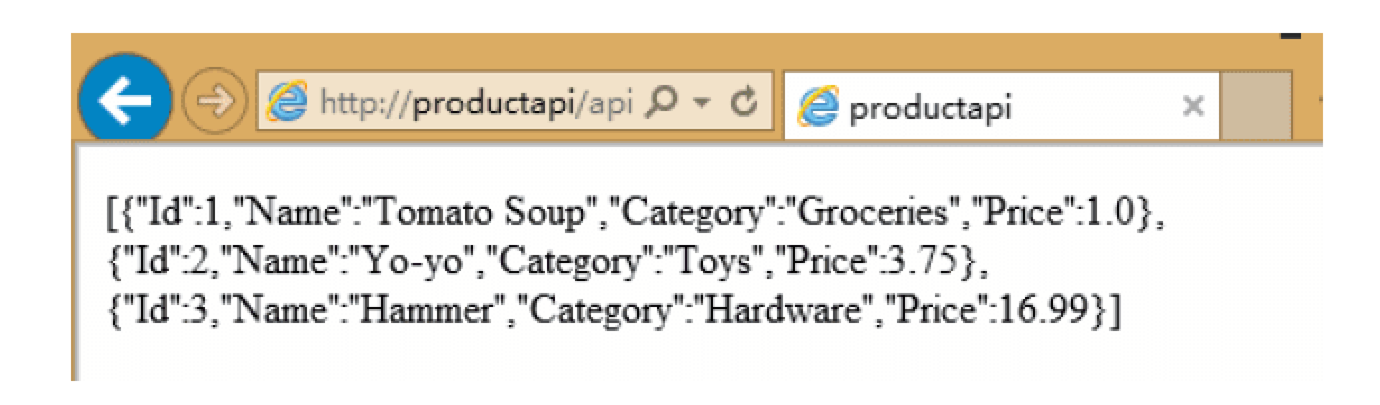


图 7-60 产品 API

下面详细演示 Azure 网站应用如何通过 Azure 混合连接访问该产品 API。

7.7.3 创建和配置 BizTalk 混合连接

（1）登录到新的 Azure 门户网站 https://portal.azure.com。新版本 Azure 门户网站还在测试中，目前只提供部分功能和英文版本。

（2）在左侧导航栏点击 Browse，然后选择 Websites。

（3）在网站列表中，选择需要访问企业内部资源的网站，在本例中，使用 ContosoMarket 网站。

（4）如图 7-61 所示，在 ContosoMarket 网站的配置页面，在 Operations 区域，单击 Hybrid connections 打开混合连接配置页面。

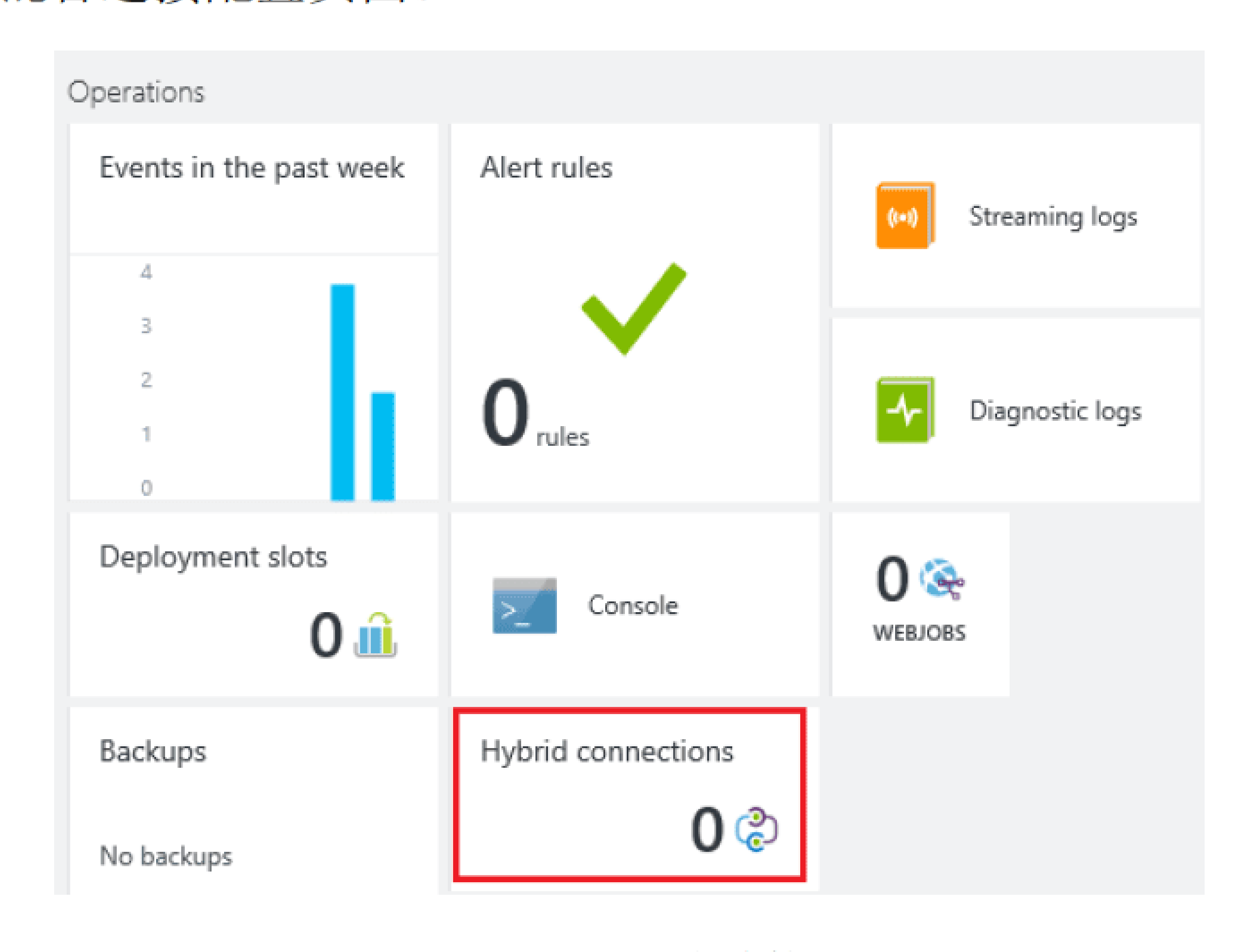


图 7-61 混合连接

（5）如图 7-62 所示，在混合连接配置页面，单击顶部的 ADD 打开添加混合连接页面。

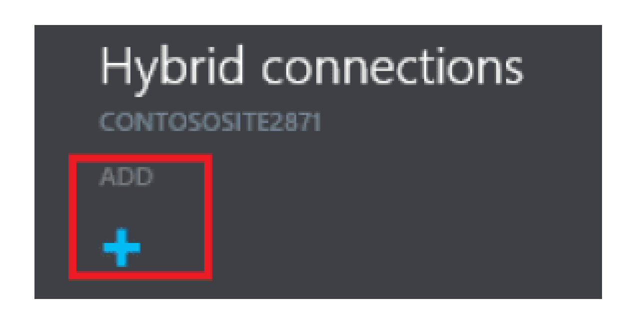


图 7-62 新建混合连接

（6）在添加混合连接页面，单击 New hybrid connection。

（7）如图 7-63 所示，在创建混合连接页面，提供如下信息：

- NAME：混合连接的名称，在这里以 ContosoProductAPI 为例。
- HOSTNAME：是指运行本地资源的主机名称。这里不一定是指机器名称。如果是通过 HTTP://ProductAPI 来访问本地资源，那么主机名指定为 productapi，端口（PORT）指定为 80。

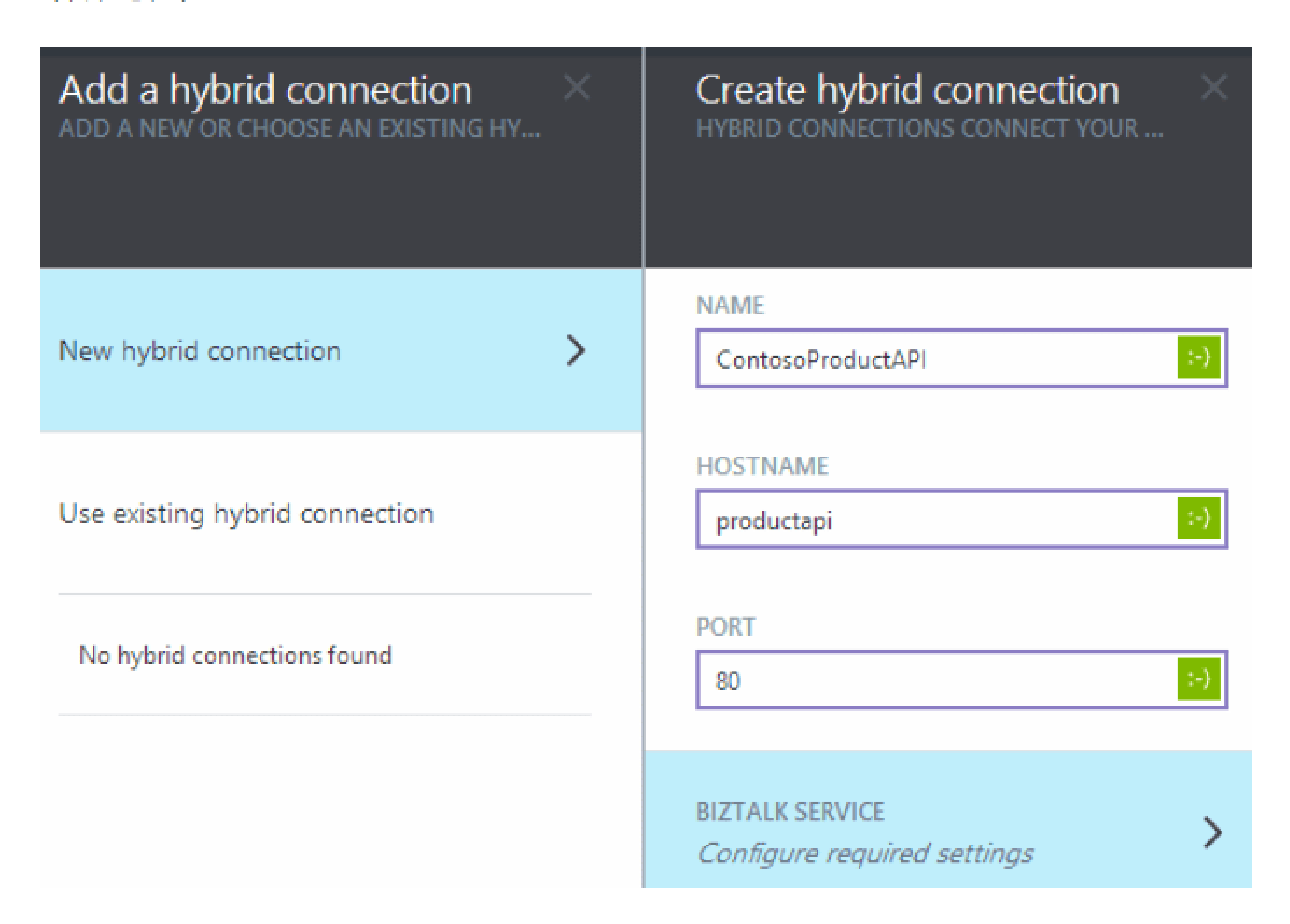


图 7-63　指定混合连接名称

（8）单击 Configure required settings，打开 Create BizTalk Service 页面。

（9）如图 7-64 所示，在 Create BizTalk Service 页面，提供服务名称，并指定数据中心等信息。

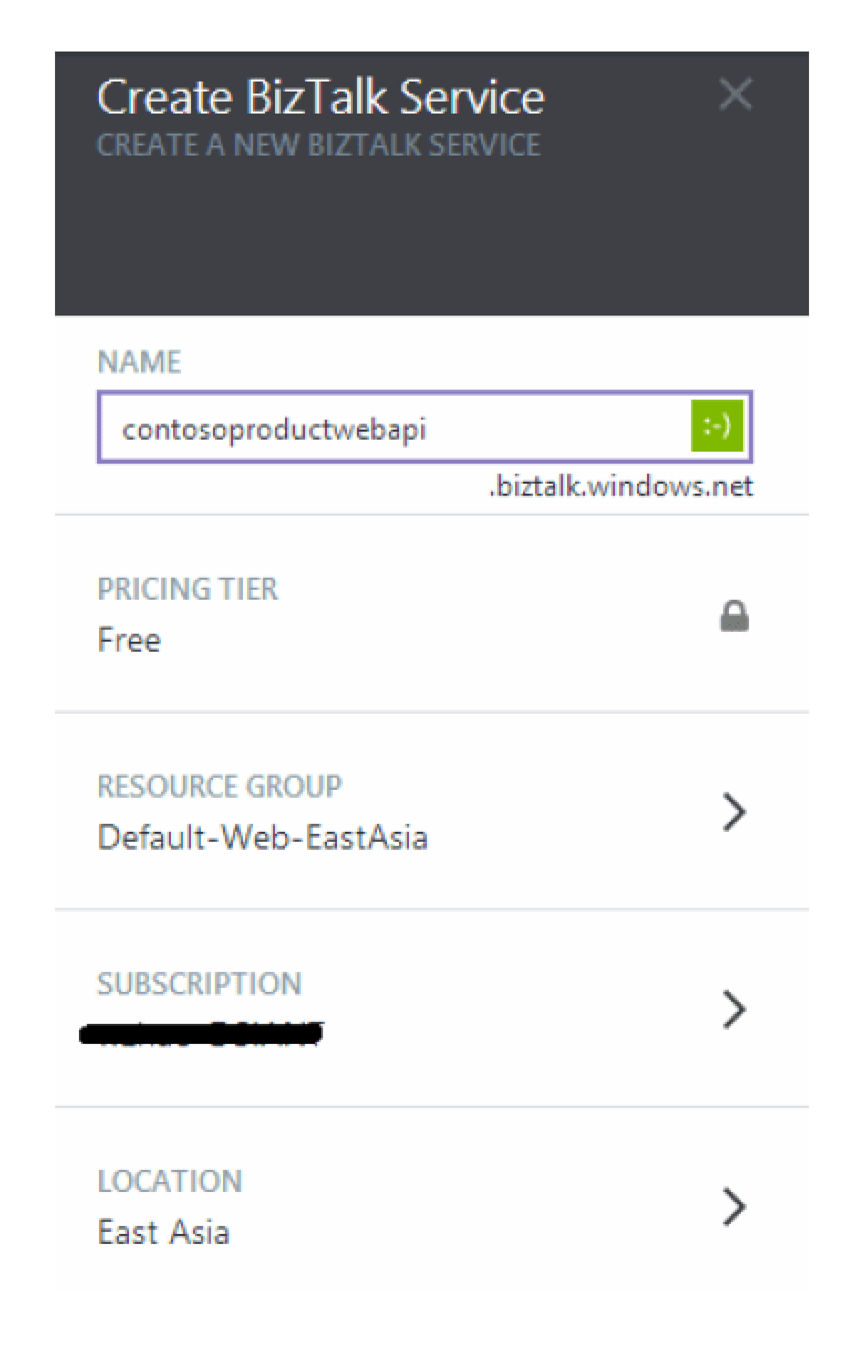


图 7-64　创建 BizTalk 服务

（10）单击 Create BizTalk Service 底部的 OK。

（11）单击 Create Hybrid Connections 底部的 OK。

（12）稍等片刻，可以看到混合连接创建成功。如图 7-65 所示，状态为 Not Connected。

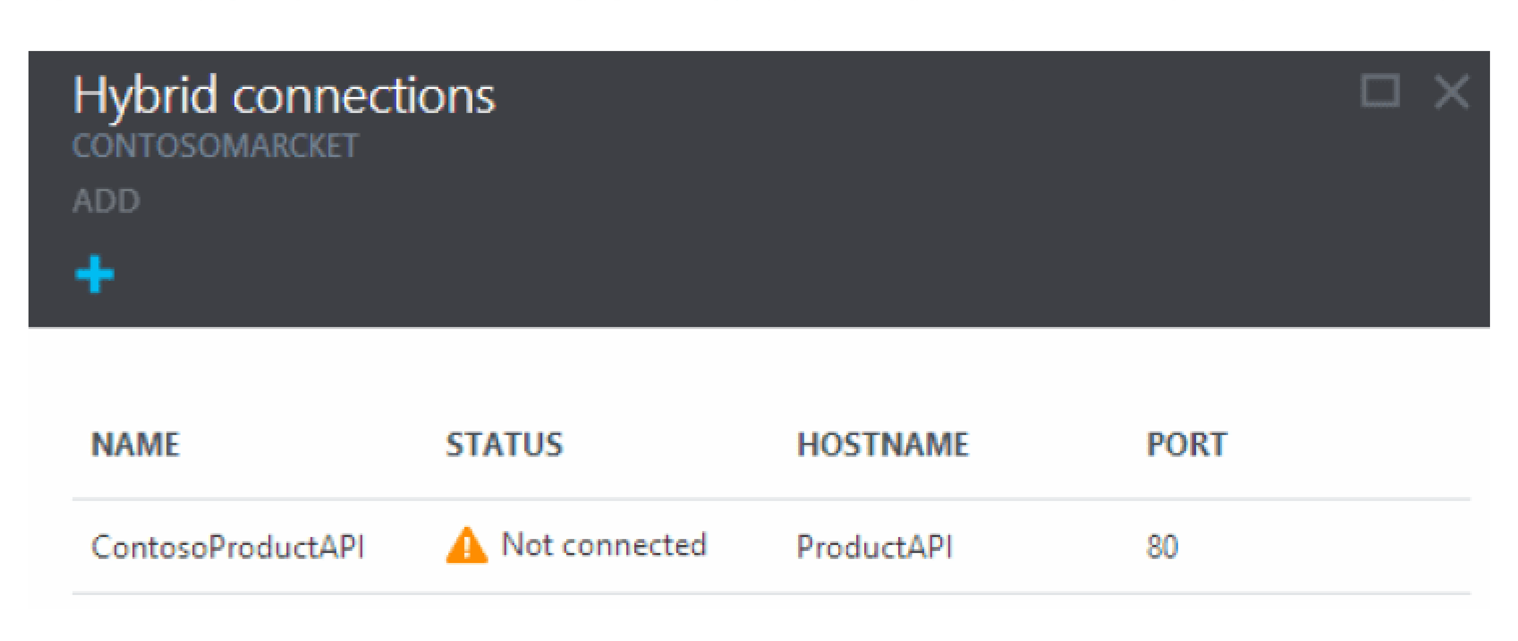


图 7-65　混合连接状态

（13）现在要配置混合连接，将它与本地资源连接起来。选择 ContosoProductAPI 混合连接，此时自动打开混合连接页面，如图 7-66 所示，单击 Listener Setup 打开混合连接属性页面。

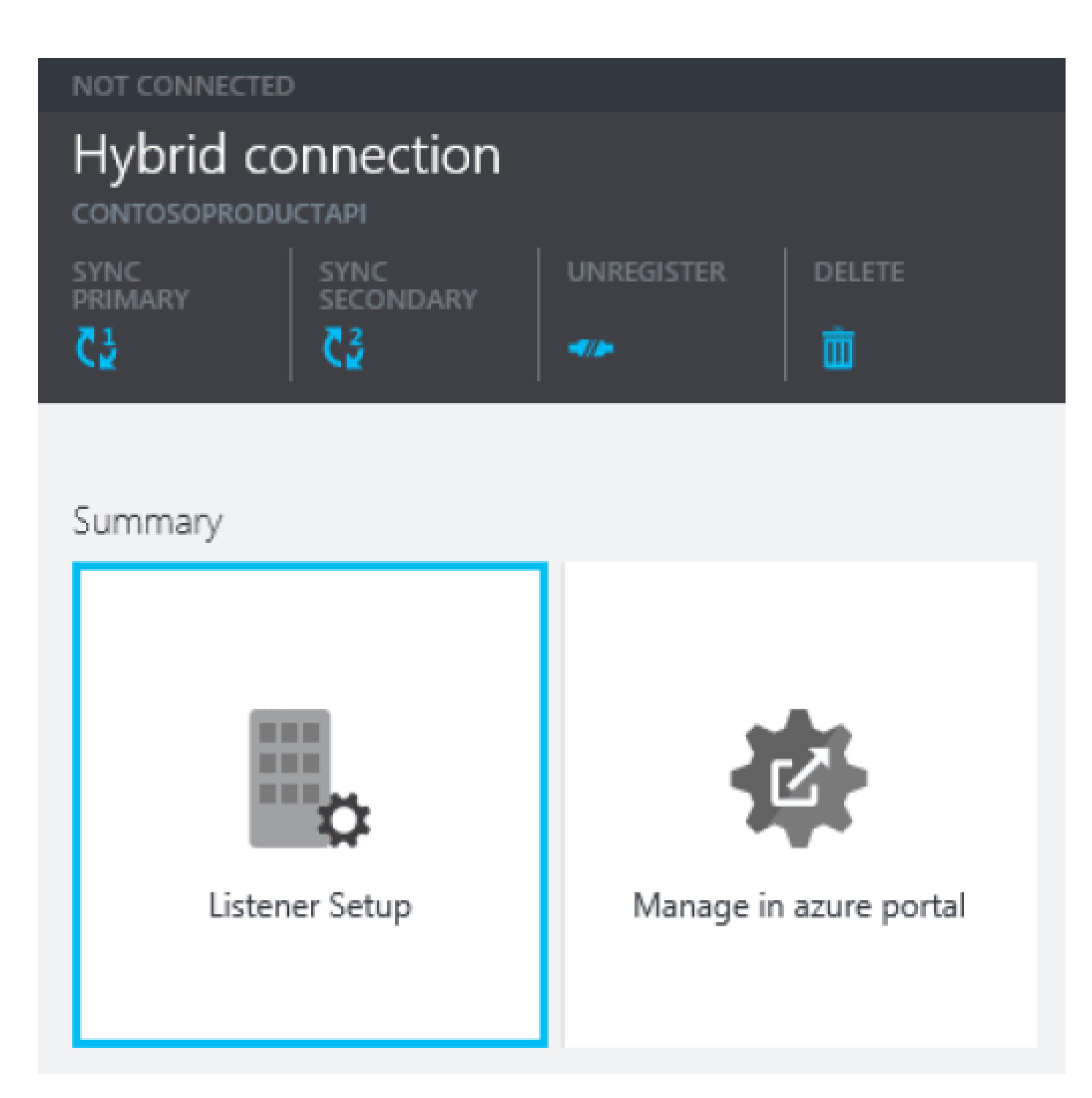


图 7-66　安装配置混合连接监听程序

（14）如图 7-67 所示，在混合连接属性页面，单击 Install and configure now。

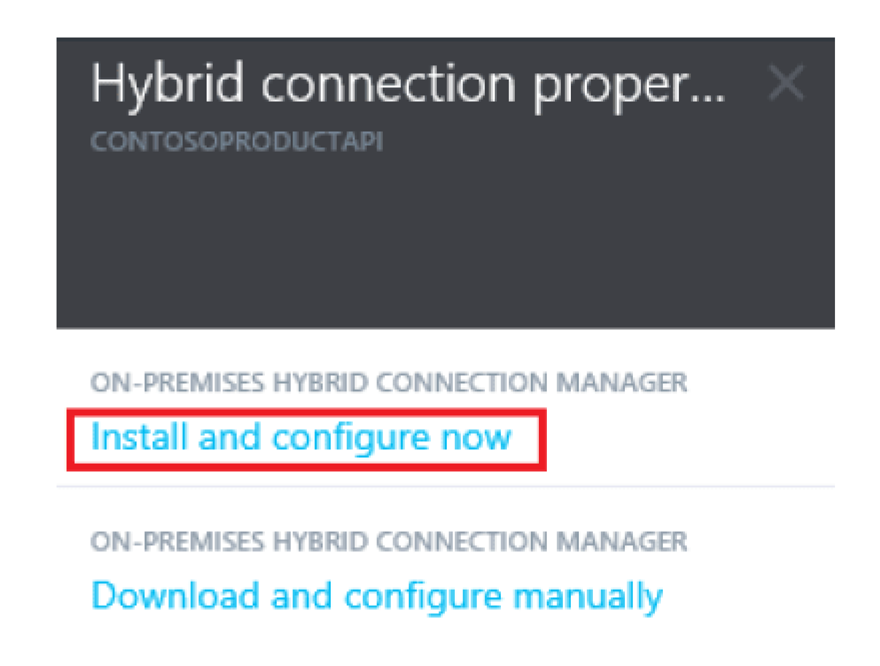


图 7-67　下载并安装混合连接管理器

（15）遵循提示安装 Microsoft Azure Hybrid Connection Manager。

（16）如图 7-68 所示，安装完成后，混合连接变为已连接（Connected）状态。

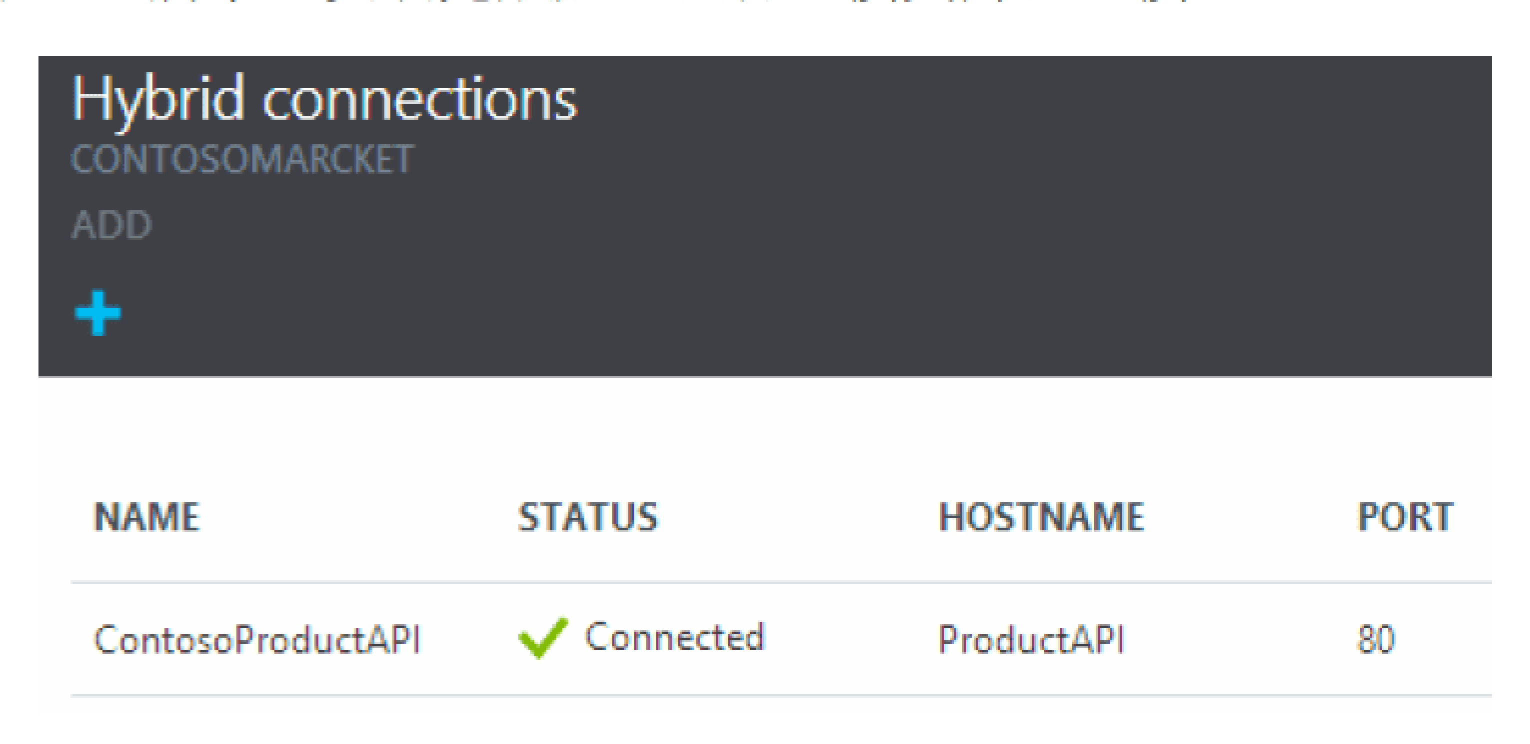


图 7-68　混合连接状态

7.7.4　开发并部署网站

7.7.4.1　示例代码

在 ContosoMarket 网站的应用中，使用了下面的代码来访问位于企业内部的产品 Web API：

```
protected async void Page Load(object sender, EventArgs e)
{
    using (var client = new HttpClient())
    {
       client.BaseAddress = new Uri("http://ProductAPI/");
       client.DefaultRequestHeaders.Accept.Clear();
       client.DefaultRequestHeaders.Accept.Add(new
           MediaTypeWithQualityHeaderValue("application/json"));
       HttpResponseMessage response = await client.GetAsync("api/products/1");
       if (response.IsSuccessStatusCode)
       {
          Product[] allProducts = await
            response.Content.ReadAsAsync<Product[]>();
          foreach (Product product in allProducts)
          {
              products.InnerHtml += "<li>" + product.Name + ": "
                  + product.Price + "</li>";
          }
        }
    }
}
```

然后，将网站应用部署到 ContosoMarket 网站，关于如何部署网站请参考前面的内容。

7.7.4.2 测试混合连接

打开浏览器，浏览网站 http://contosomarcket.azurewebsites.net/，会看到如图 7-69 所示的产品信息。



图 7-69 网站运行结果

上面的示例演示了 Azure 网站如何通过混合连接访问位于企业内部的资源。采用混合连接，应用无须任何修改、企业无须建立 VPN，也无须额外的防火墙设置，是混合云解决方案的理想选择。

7.8 Azure 网站集成虚拟网络

混合连接提供了点到点的解决方案。通过混合连接，部署在 Azure 网站中的 Web 应用可以访问位于企业内部的某个指定的资源，比如数据库、Web 服务等。对于大型企业而言，更希望站点到站点的解决方案。比如，企业内部数据中心的网络和位于 Azure 中的资源可以互通。通过 Azure 虚拟网络，IT 管理员可以创建站点到站点的混合云解决方案。

Azure 虚拟网络是客户管理的位于 Azure 内部的私有虚拟网络。通过虚拟网络，IT 管理员可以实现以下目标。

1. 创建仅限于 Azure 云中的虚拟网络

IT 管理员可以将运行在 Azure 中的虚拟机（Azure VM）和云服务（cloud service）加入到 Azure 虚拟网络中。在 Azure 虚拟网络中的虚拟机和云服务可以安全地互相直接访问，而无须经过 Internet。

在本书开始介绍了 Azure 网站适用于传统的两层网站架构：Web 层和数据库。Azure 云服务适用于 3 层网站架构：Web 层、业务逻辑层和数据库。如图 7-70 所示，通过点到站点（P2S）VPN，可以将 Azure 网站与 Azure 虚拟网络集成起来，从而将 Azure 网站扩展到 3 层网站架构。把网站和运行业务逻辑以及数据库的虚拟机加入到虚拟网络后，Web 应用可以直接使用内部 IP 访问业务逻辑层而无须经过 Internet，在提高安全性的同时大大提高了性能。

2. 安全的扩展企业数据中心

如图 7-71 所示，通过站点到站点（Site-to-Site）VPN，可以将 Azure 虚拟网络与本地

数据中心互联。站点到站点 VPN 使用行业标准的 IPSec 协议在企业数据中心的网关与 Azure 虚拟网络之间提供安全连接。站点到站点 VPN 完美实现了混合云解决方案，位于 Azure 虚拟网络中的虚拟机和服务可以安全地访问位于企业数据中心的任何资源，譬如数据库、Web 服务和 UNIX 系统等。另外，站点到站点 VPN 安全扩展了本地数据中心的容量和规模。

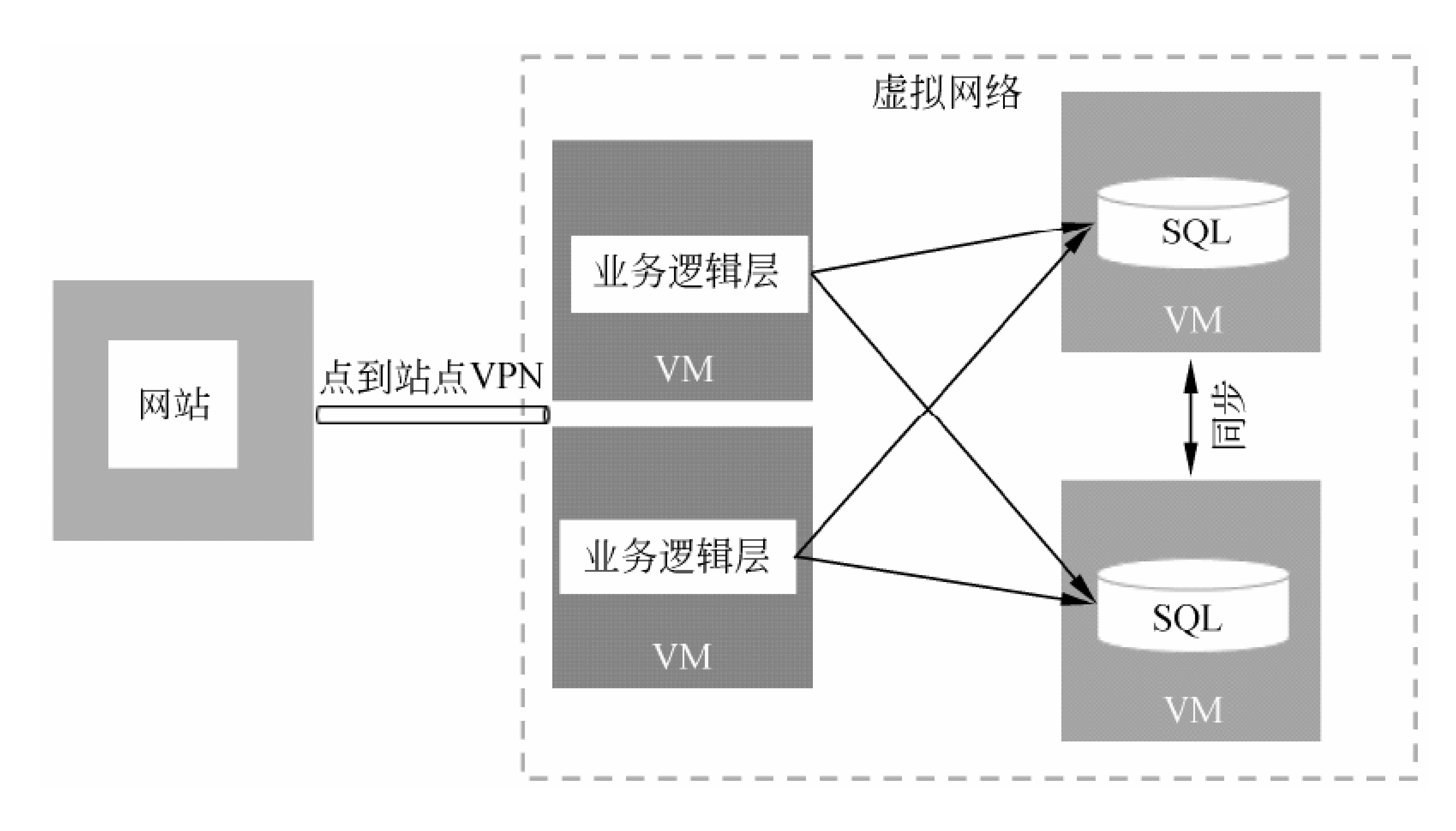


图 7-70　通过虚拟网络实现典型的 3 层网站架构

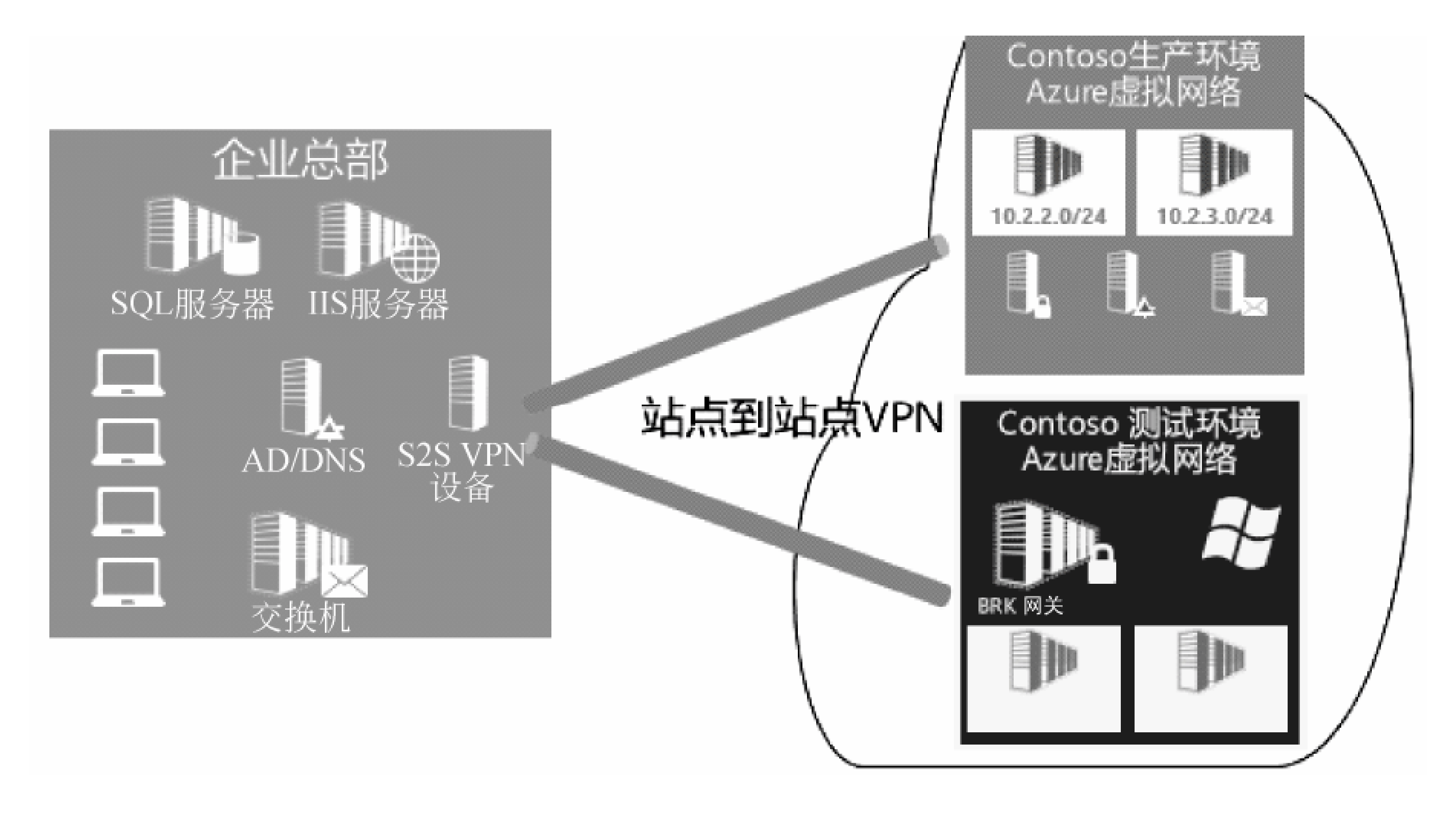


图 7-71　站点到站点 VPN（虚拟专用网络）

7.8.1　创建虚拟网络

（1）登录到 Azure 管理门户网站。

（2）在屏幕左下角，单击“新建”。在导航窗格中，单击“网络服务”，然后单击“虚拟网络”，单击“自定义创建”以开始配置向导。

（3）在“虚拟网络详细信息”页面中，输入以下信息，然后单击右下方的“下一步”箭头。

- 名称：命名你的虚拟网络。例如 ContosoVnet。
- 位置：指定虚拟网络所在的 Azure 数据中心。所有部署到此虚拟网络的资源（虚拟机）的物理位置都将位于指定的数据中心。例如，加入到位于香港数据中心的某个虚拟网络的所有虚拟机都位于香港数据中心。创建虚拟网络后，将无法更改与虚拟网络关联的区域。

（4）在“DNS 服务器和 VPN 连接”页面上，输入以下信息，然后单击右下方的“下一步”箭头。

- DNS 服务器：输入 DNS 服务器名称和 IP 地址，或者从下拉列表中选择一个以前注册的 DNS 服务器。此设置不创建 DNS 服务器，但可以指定要用于对此虚拟网络进行名称解析的 DNS 服务器。如果你希望使用 Azure 默认的名称解析服务，请将本部分留空。本例中选择使用 Azure 默认的名称解析服务。
- 配置点到站点 VPN：选中该复选框。

（5）在“点到站点连接”页面上，指定 VPN 客户端在连接到 VPN 后的 IP 地址范围。必须确保指定的地址范围不与本地网络上的任何范围相重叠。IP 地址数量与 VNET 中需要的虚拟机数量有关。在本例中，只需要一台虚拟机，因此可用 IP 地址范围只需满足该需求即可。如图 7-72 所示，输入以下信息，然后单击“下一步”箭头。

- 地址空间：包括“起始 IP”和 CIDR（地址数）。
- 添加地址空间：仅在网络设计需要时添加。

地址空间	起始 IP	CIDR (地址数)	可用的地址范围
10.0.0.0/29	10.0.0.0	/29 (6)	10.0.0.1 - 10.0.0.6

图 7-72　指定点到站点（P2S）VPN 地址范围

（6）在“虚拟网络地址空间”页面中，指定要用于虚拟网络的地址范围。这些都是动态 IP 地址（DIP），这些 DIP 将分配部署到此虚拟网络的虚拟机和其他资源。所选范围不要与本地网络所用地址范围重叠，这一点尤其重要。

（7）输入以下信息，然后单击创建图标开始创建虚拟网络。

- 地址空间：添加希望用于此虚拟网络的内部 IP 地址范围，包括“起始 IP”和“地址数”。
- 添加子网：附加的子网不是必需的，但可能需要为具有静态 DIP 的虚拟机创建一个单独的子网。
- 添加网关子网：网关子网是点到站点 VPN 必需的。网关子网仅用于此虚拟网络网关。

（8）创建虚拟网络后，将看到在管理门户网站中“网络”页面上的“状态”下列出“已创建”。此时即可创建动态路由网关。

（9）创建动态路由网关。

① 在管理门户中的“网络”页面上，单击刚刚创建的虚拟网络，然后导航到“仪表板”页面。如图 7-73 所示，显示状态为“未创建网关”。

② 单击位于“仪表板”页面底部的“创建网关”。创建网关大约需要 15 分钟。



图 7-73　网络状态

③ 网关创建成功后，如图 7-74 所示，在虚拟网络的“仪表板”页面显示网关 IP 地址。



图 7-74　网关 IP 地址

7.8.2　新建虚拟机并加入虚拟网络

（1）登录到 Azure 管理门户网站。

（2）在左下角单击“新建”，依次选择“计算”→“虚拟机”→“从库中”。

（3）在“选择映像”对话框，选择 Windows Server 2012 R2 Datacenter，单击“下一步”。

（4）在“创建虚拟机”页面，如图 7-75 所示，指定虚拟机名称和管理员用户名及密码，然后单击“下一步”。



图 7-75　配置虚拟机

（5）如图 7-76 所示，指定虚拟机的配置。

- 云服务 DNS 名称：在“云服务”列表中，选择“创建新云服务”。Azure 虚拟机基于 Azure 云服务，默认使用虚拟机名称作为对应的云服务名称。
- 存储账户：可以选择新建存储账户或者使用已有的存储账户，此处选择已经存在的存储账户。
- 虚拟网络子网：选择在 7.8.1 节创建的 ContosoVnet 虚拟网络。



图 7-76　指定虚拟机配置

（6）单击“下一步”，选择需要安装的扩展，比如安全软件。

（7）最后单击“完成”，创建虚拟机。

（8）虚拟机创建完成后，在管理门户中的“网络”页面上，单击刚刚创建的虚拟网络，然后导航到“仪表板”页面。如图 7-77 所示，新创建的虚拟机出现在虚拟网络的资源列表中。



名称	角色	IP 地址	子网名称
ContosoRedisSrv	虚拟机	10.0.0.68	Subnet-1

图 7-77　虚拟网络资源列表

7.8.3　安装 Redis Cache

（1）登录到 Azure 管理门户网站。

（2）在左侧导航栏选择“虚拟机”，然后选择 7.8.2 节创建的 ContosoRedisSrv 虚拟机。

（3）在底部命令栏，单击“链接”。输入用户名和密码远程登录到 ContosoRedis 虚

拟机。

（4）访问 http://nuget.org/nuget.exe，下载 NuGet。

（5）打开命令行控制台，并执行下面的命令安装 Windows 版的 NuGet：

```
nuget install redis-64
```

（6）NuGet 默认将 Redis 安装在当前文件夹，从命令行中进入该文件夹，执行下面的命令安装 Redis 服务：

```
redis-server --service-install redis.windows.conf
```

（7）执行下面的命令启动 Redis 服务：

```
redis-server --service-start
```

7.8.4　配置 Redis 虚拟机端点

（1）登录到 Azure 管理门户网站。

（2）在左侧导航栏选择“虚拟机”，然后选择 7.8.2 节创建的 ContosoRedis 虚拟机。

（3）单击“端点”，导航到“端点配置”页面。

（4）单击底部的“添加”命令按钮，添加新的端点。

（5）在“将端点添加到虚拟机”页面，选择“添加独立终结点”，单击“下一步”。

（6）在“指定端点的详细信息”页面，如图 7-78 所示，添加 Redis 服务的端口 6379。单击“完成”按钮开始创建。



图 7-78　指定端点详细信息

7.8.5　将 Azure 网站通过 VPN 连接到虚拟网络

（1）登录到新的测试版 Azure 管理门户网站 https://portal.azure.com。当前管理门户网

站暂时不支持该功能。

（2）在左侧命令栏，单击“浏览”，选择“网站”。

（3）在网站列表中选择需要连接到 VPN 的网站，并打开该网站。注意：只有标准模式网站支持连接到 VPN。在该示例中，使用 ContosoVPNSite。

（4）在“网站配置”页面的“网络”区域，如图 7-79 所示，显示“此站点未连接到虚拟网络”。单击该区域打开“虚拟网络”配置页面。



图 7-79　未连接到虚拟网络

（5）在“虚拟网络配置”页面，选择“使用现有虚拟网络”。在虚拟网络列表中选择 ContosoVnet。单击“确认”。

（6）VPN 连接成功后，如图 7-80 所示，显示虚拟网络的名称。

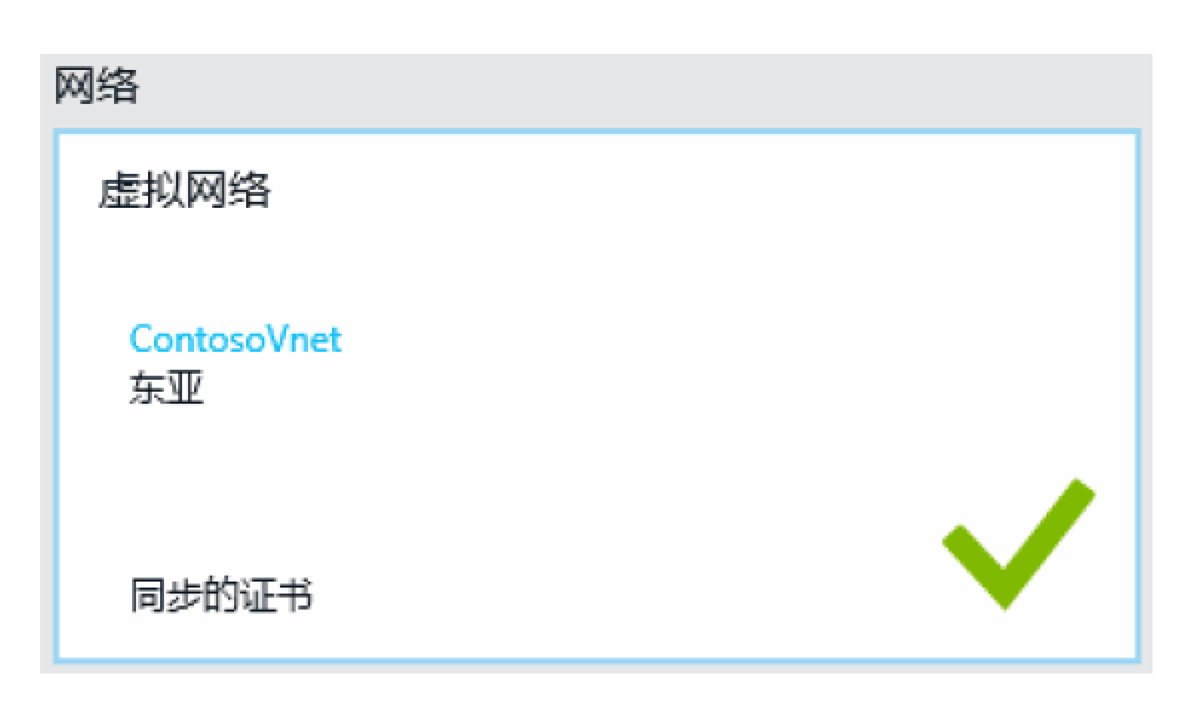


图 7-80　连接到虚拟网络

（7）登录到当前 Azure 管理门户网站 https://manage.windowsazure.com，在“网络”页面上，单击之前创建的虚拟网络，然后导航到“仪表板”页面。如图 7-81 所示，显示客户端数量为 1。这表明网站已经通过 VPN 连接到了 ContosoVnet 虚拟网络。

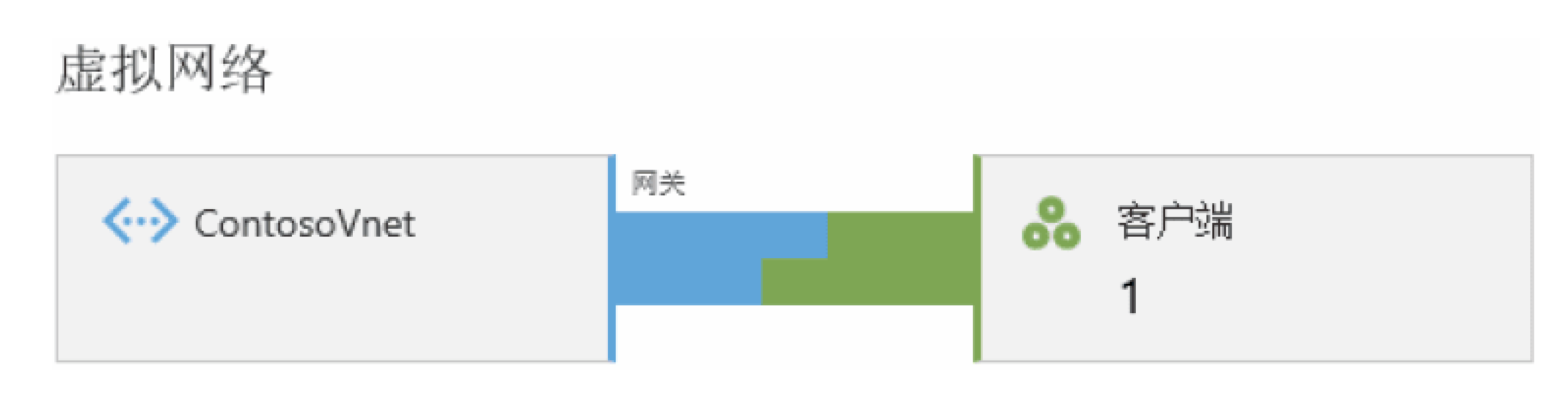


图 7-81　VPN 客户端

（8）导航到“证书”页面，如图 7-82 所示，可以看到 VPN 证书已经自动配置成功。



图 7-82　VPN 证书

7.8.6　在 Azure 网站应用中使用 RedisCache

（1）在 Visual Studio 中，选择“文件”菜单，然后选择“新建”→“项目”。

（2）在“新建项目”对话框中，选择 Visual C#，然后选择 Web。给定项目名称，在本例中命名为 RedisCache，单击“确定”按钮。

（3）在“新建 ASP.NET 项目”对话框中，选择 Empty，单击“确定”按钮创建空的项目。

（4）在解决方案管理器中，右击 RedisCache，选择“添加”，然后选择“添加 Web 窗体”。

（5）在“指定项名称”对话框中，将页面命名为 Default。

（6）单击“工具”菜单，选择“NuGet 程序包管理器”→“程序包管理器控制台”。

（7）在程序包管理器控制台中，运行 Install-Package StackExchange.Redis 命令安装 StackExchange 的 Redis 客户端。

（8）打开 Default.cs 文件，在引用部分加入如下代码：

```
using StackExcange.Redis;
```

（9）在 Page_Load 函数中加入如下代码，使用虚拟机的内部 IP 地址，而不是公网 IP 地址。

```
protected void Page_Load(object sender, EventArgs e)
{
    ConnectionMultiplexer connection = ConnectionMultiplexer.Connect
    ("10.0.0.68");
    IDatabase cache = connection.GetDatabase();
    var now = cache.StringGet("CurrentTime");
    if (now == RedisValue.Null)
    {
        now = DateTime.UtcNow.ToString();
        cache.StringSet("CurrentTime", now);
    }
    this.now.InnerText = "Current Time: " +  DateTime.UtcNow.ToString();
    this.cachedTime.InnerText = "Cached time: " + now;
}
```

7.8.7 测试网站 VPN 连接

将 RedisCache 项目部署到 Azure 网站后，访问 ContosoVPNSite。如图 7-83 所示，可以看到缓存的时间与当前时间的区别。

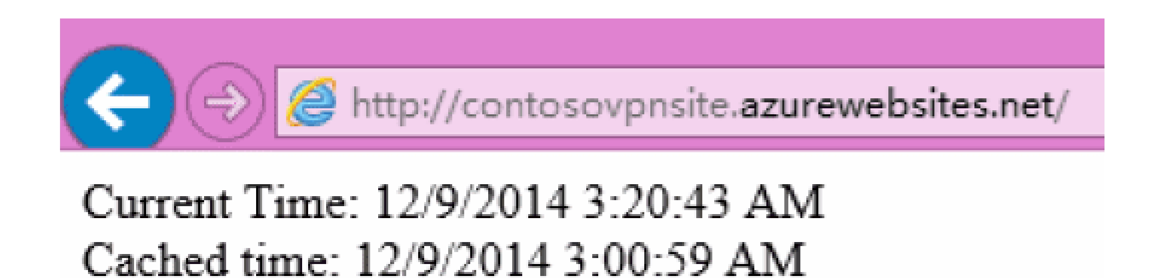


图 7-83 缓存在 Redis 里的时间

7.9 利用 Web 作业执行后台任务

7.9.1 Web 作业简介

Web 作业（Web job）是 Azure 网站提供的执行后台任务的功能。通过 Web 作业，能够在与网站相同的上下文中运行程序或脚本。Web 作业可以在后台运行一些需要较长时间处理的任务，从而可以提高网页的响应速度。

下面是一些适用于 Web 作业的典型场景：

- 图片、视频文件处理或其他大量占用 CPU 的工作。很多网站（比如社交网站、个人网站）的共同特点是需要客户上传大量的图像或视频。通常，在内容上传之后，需要对媒体文件进行一些操作，比如内容检测、生成缩略图、添加水印等。但是，这些操作非常耗时，会降低用户体验。通过 Web 作业，可以在后台执行这些任务。
- 队列处理。Web 前端与后端服务进行异步通信的常见方式是使用队列。网站只需要将后台任务推送到队列中，而无须等待任务完成，从而提高网页的响应速度。后端的 Web 作业从队列中读取任务并处理。例如，可以将数据库插入操作放到队列中而不是直接写入 SQL 数据库，让后端 Web 作业处理数据库的工作。
- 文件维护，比如清理日志文件。Azure 网站的文件空间有限，您可能希望一个计划任务定期清理网站的日志文件。比如，将不需要的文件删除，将需要归档的文件上传到 Azure 存储中备份保存等。
- 运行时间较长的任务。比如在购物季节开始前向所有顾客发送促销电子邮件。
- 定期运行的任务。比如每周或者每月固定生成商业报表等。

7.9.2 Web 作业类型

任何可执行程序都可以作为 Web 作业，比如：

- .exe（非 UI 程序），.cmd，.bat。

- PowerShell 脚本。
- .js（通过 Node.js 运行）。
- .php。
- .py。
- .sh（使用 bash）。

Azure 网站首先查找命名为 run 的文件，比如 run.exe、run.bat 等。扩展名的优先顺序为.cmd、.bat、.exe、.ps1、.sh、.php、.py、.js。如果同时有 run.exe 和 run.bat，那么 run.bat 的优先级高于 run.exe，因此 run.bat 会被执行。

如果没有命名为 run 的文件，则用扩展名查找。首先查找*.cmd 文件，如果找到则执行，否则继续查找*.bat、*.exe 等，以此类推。

Web 作业有两种类型：触发式（trigger）和连续式（continuous）。其中触发式又分为按需运行模式和计划模式。触发式在任务执行完成后即退出，而连续式类似于 Windows 的服务程序，在后台持续运行。

7.9.2.1　按需运行模式

按需运行模式为手动模式。在该模式下，需要通过管理门户网站或者 Azure PowerShell、Azure X-CLI 等方式手动运行 Web 作业。

下面的步骤创建一个按需运行的 Web 作业。

（1）登录到 Azure 管理门户网站，打开要创建 Web 作业的网站。

（2）单击顶部的 Web 作业打开 Web 作业配置页面。

（3）单击底部命令栏的“创建”按钮。

（4）如图 7-84 所示，在“内容”一栏，选择要作为 Web 作业运行的程序或脚本（必须为压缩包）。



图 7-84　创建按需运行的 Web 作业

（5）在“如何运行”一栏选择“按需运行”。

（6）单击窗口右下角的“完成”按钮，创建 Web 作业。

（7）当需要运行 Web 作业时，单击底部命令栏的“运行一次”按钮。

7.9.2.2 计划模式

在计划模式下， Web 作业在指定的时间被运行。可以指定 Web 作业运行一次或者是周期性运行。Azure 计划程序负载在指定的时间运行 Web 作业。

下面的步骤创建一个计划模式的 Web 作业，该 Web 作业每周执行一次。

（1）登录到 Azure 管理门户网站，打开要创建 Web 作业的网站。

（2）单击顶部的“Web 作业”打开“Web 作业配置”页面。

（3）单击底部命令栏的“创建”按钮。

（4）如图 7-85 所示，在“内容”一栏，选择要作为 Web 作业运行的程序或脚本（必须为压缩包）。



图 7-85 创建按时间表运行的 Web 作业

（5）在“如何运行”一栏选择“按时间表运行”。

（6）在“计划程序区域”一栏选择希望的数据中心，通常与网站在相同的数据中心。

（7）单击窗口右下角的“下一步”按钮，定义时间表。

（8）如图 7-86 所示，重复周期有两个选择：一次性作业和定期作业。在这里，选择“定期作业”。执行间隔设定为 7 天。最后设置结束时间，这里设置为 2050 年 10 月 4 日 0 时。



图 7-86 定义 Web 作业运行时间表

（9）单击右下角的“完成”按钮，创建按计划模式运行的 Web 作业。

7.9.2.3　连续运行模式

连续运行模式类似于 Windows 服务，Web 作业始终处于运行状态。

在 Azure 网站中，如果网站在 20 分钟内没有用户访问，那么网站会被关闭。当用户再次访问时会被重新创建。在连续运行模式下，如果网站被关闭，那么 Web 作业也被关闭。这可能会中断正在运行的任务，因此，如果选择连续运行模式，建议打开始终可用（Always On）功能。

如果网站运行在多个机器实例上，那么连续运行的 Web 作业也会运行在每个机器实例上。如果只希望运行一个 Web 作业实例，那么需要在 Web 作业的根目录创建一个 settings.job 文件，该文件需要包含如下内容：

```
{"is_singleton": true}
```

下面的步骤创建一个连续运行的 Web 作业。

（1）登录到 Azure 管理门户网站，打开要创建 Web 作业的网站。

（2）单击顶部的“Web 作业”打开“Web 作业配置”页面。

（3）单击底部命令栏的“创建”按钮。

（4）如图 7-87 所示，在“内容”一栏，选择要作为 Web 作业运行的程序或脚本（必须为压缩包）。



图 7-87　创建连续运行的 Web 作业

（5）在“如何运行”一栏选择“连续运行”。

（6）单击窗口右下角的“完成”按钮，创建 Web 作业。

（7）当需要运行 Web 作业时，单击底部命令栏的“运行一次”按钮。

7.9.3　Web 作业部署

Web 作业的部署非常简单，只需要将文件复制到指定的目录即可。可以通过管理门户网站、FTP、Kudu 控制台以及 Visual Studio 来部署。

与网站文件相同，Web 作业也保存在 wwwroot 目录下。目录的结构如下：

/wwwroot/app_data/jobs/<job_type>/<job-name>/<script-content>

其中，job_type 为作业类型。

通过 FTP 可以查看 Web 作业的目录结构。图 7-88 描述了先前创建的按需运行模式的 Web 作业的目录结构。

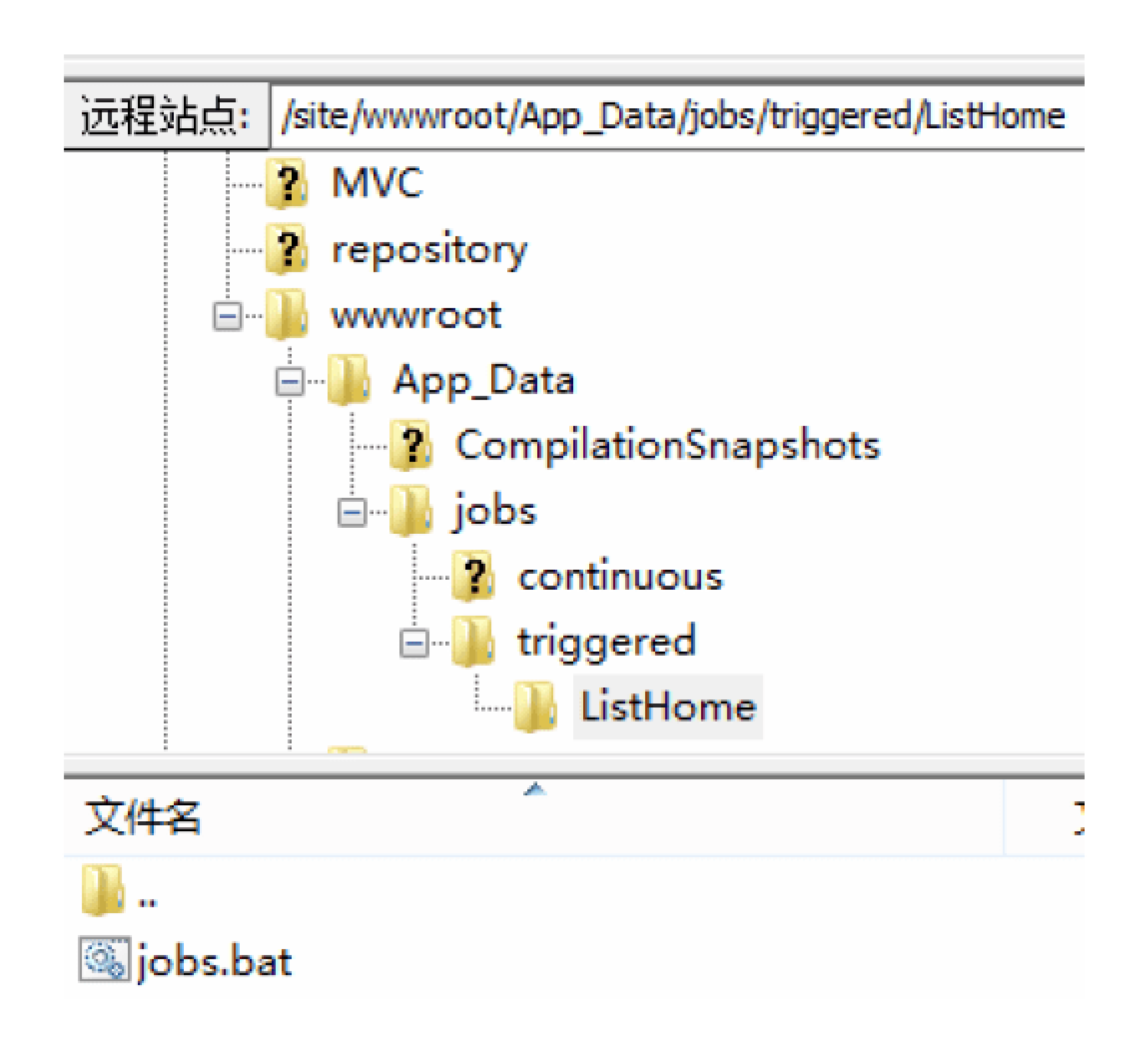


图 7-88　Web 作业文件目录结构

图 7-89 描述了连续运行作业的文件目录结构。

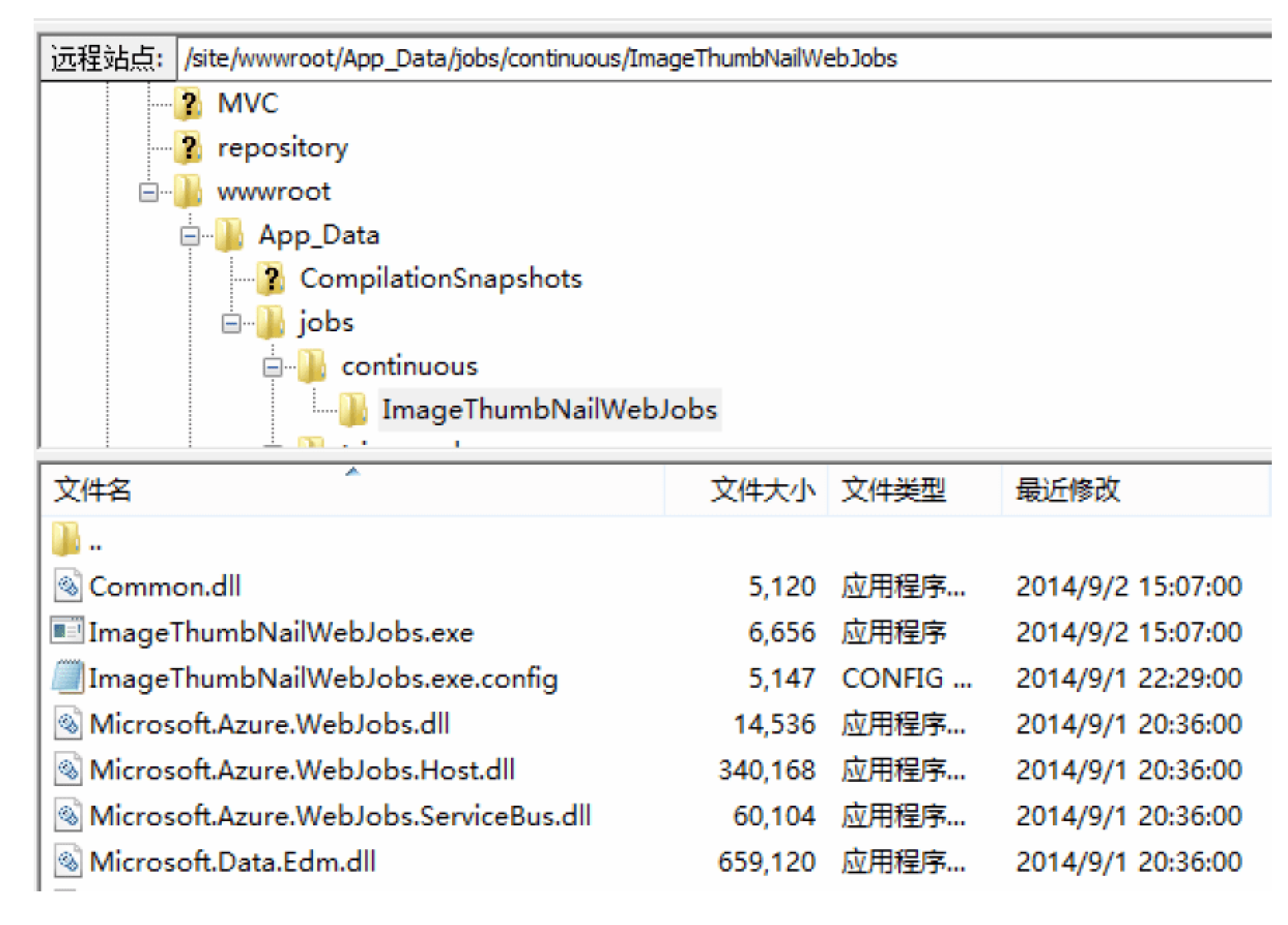


图 7-89　连续运行 Web 作业目录结构

Web 作业运行时产生的日志则保存在/data 目录下，目录结构为：

/data/jobs/<job_type>/<job-name>/<run-id>/

图 7-90 描述了 Web 作业运行日志目录的结构。

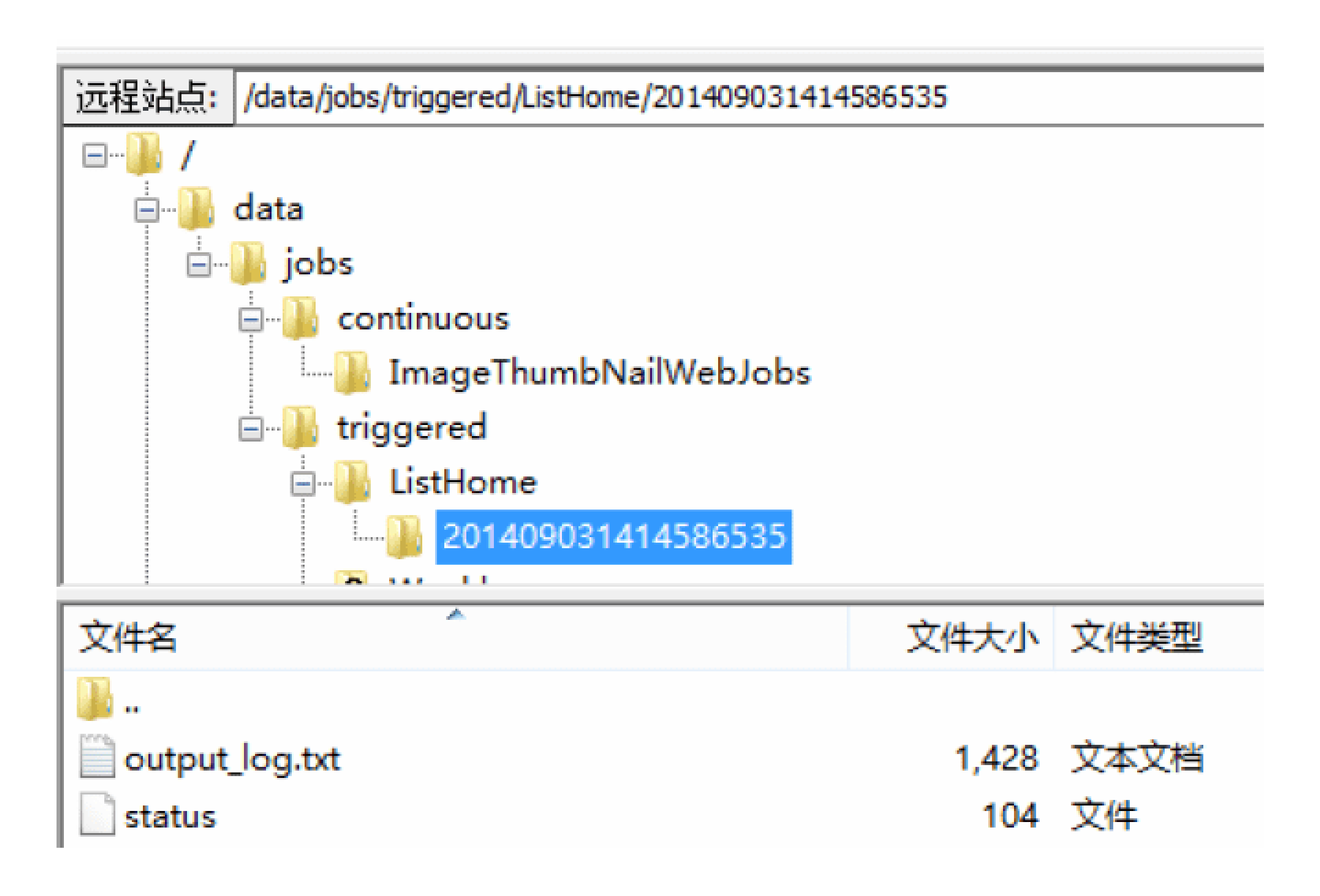


图 7-90　Web 作业日志目录结构

7.9.4　Web 作业实例

在前面的章节中介绍过，Azure 网站中，每个网站可用的文件空间是有限的。具体的限额请参考下面的文档：

http://azure.microsoft.com/en-us/pricing/details/websites/

很可惜的是，经常遇到客户的网站因为文件存储超过限额而影响业务的情况。通常这些是由于过多的日志文件占用了太多的空间。下面编写一个小程序，定期发送邮件给管理员报告网站文件空间使用情况。

7.9.4.1　创建 Web 作业应用

首先，在 Visual Studio 中创建一个 C#控制台应用程序。在 Main 函数中加入下面的代码。Main 函数首先调用 GetDirectorySize 函数获取网站的磁盘空间使用情况，然后调用 SendMail 函数发送邮件给管理员。

```
static void Main(string[] args)
{
    string folder = Environment.GetEnvironmentVariable("HOME");
    long size = GetDirectorySize(folder);
    string siteName = Environment.GetEnvironmentVariable("WEBSITE SITE NAME");
    string mailSubject = siteName + " disk usage";
    string mailBody = "At " + DateTime.UtcNow.ToString()
              + ", your site disk usage is: "
              + size.ToString()
              + " bytes ";
    SendMail("Admin's mail", mailBody, mailSubject);
}
```

GetDirectorySize 函数的实现非常简单，只包含下面的两行代码：

```
private static long GetDirectorySize(string folderPath)
        {
            DirectoryInfo di = new DirectoryInfo(folderPath);
            return di.EnumerateFiles("*.*", SearchOption.AllDirectories).
            Sum(fi => fi.Length);
        }
```

SendMail 函数利用.NET 的 SmtpClient 类来发送邮件。它利用了 Windows Live 的 SMTP 服务器，要求有一个 Windows Live 或者 Hotmail 的账号。

```
private static void SendMail(string receiver, string body, string subject,
bool isHtml=false)
{
    SmtpClient SmtpServer = new SmtpClient("smtp.live.com");
    SmtpServer.Port = 587;
    SmtpServer.UseDefaultCredentials = false;
    SmtpServer.Credentials = new System.Net.NetworkCredential("Sender's
    Live Mail", "password");
    SmtpServer.EnableSsl = true;
    var mail = new MailMessage();
    mail.From = new MailAddress("Sender's Live Mail");
    mail.To.Add(receiver);
    mail.Subject = subject;
    mail.IsBodyHtml = isHtml;
    mail.Body = body;
    SmtpServer.Send(mail);
}
```

7.9.4.2 部署 Web 作业

现在部署 Web 作业到 Windows Azure 网站。在 Microsoft Azure SDK for .NET 2.4 中提供了直接从 Visual Studio 中部署 Azure Web 作业的功能。如图 7-91 所示，在 Visual Studio 2013 的解决方案资源管理器中，右击该项目，然后选择“发布为 Azure Webjob…”。

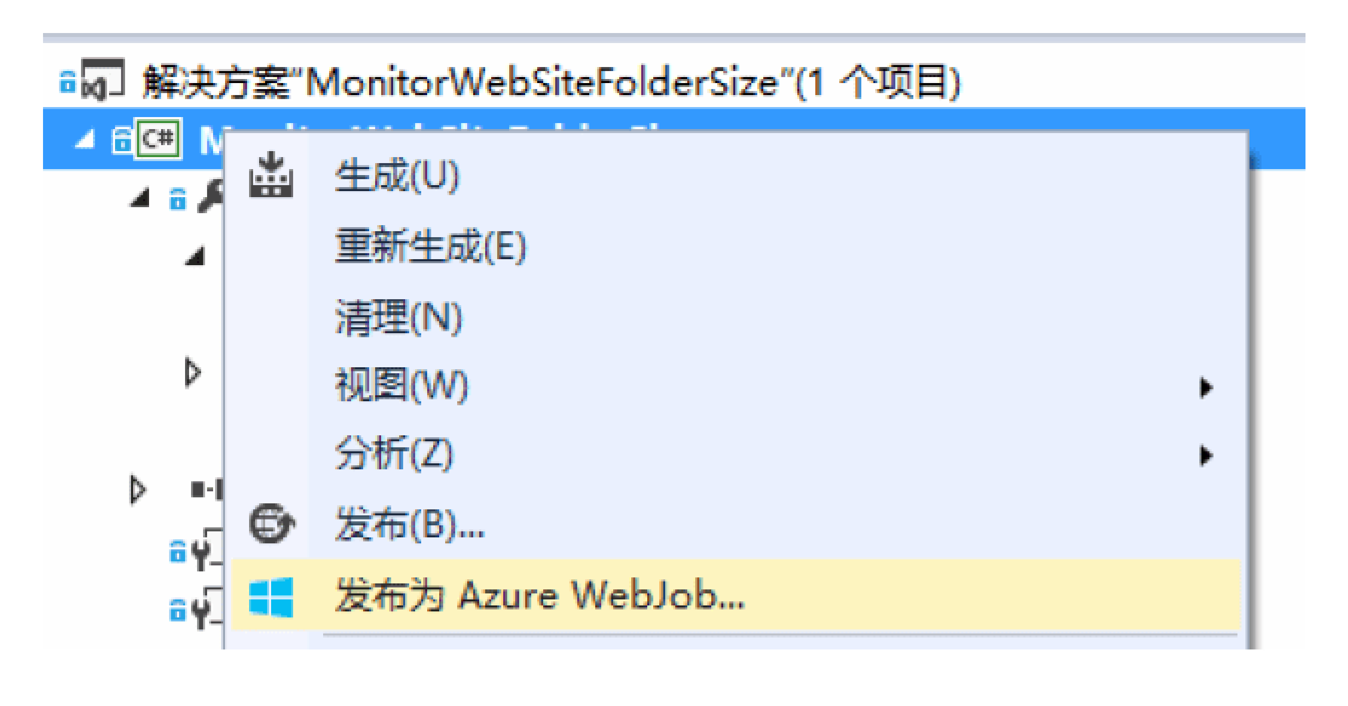


图 7-91　从 Visual Studio 中发布 Web 作业

如图 7-92 所示，在“添加 Azure WebJob”对话框中，单击“Web Job 运行模式（R）”，

选择“重复性作业”。然后设置重复间隔，在这里，将重复间隔设置为 1 分钟。在生产环境中，建议根据需要合理设置重复间隔。过于频繁地运行可能会影响网站性能。

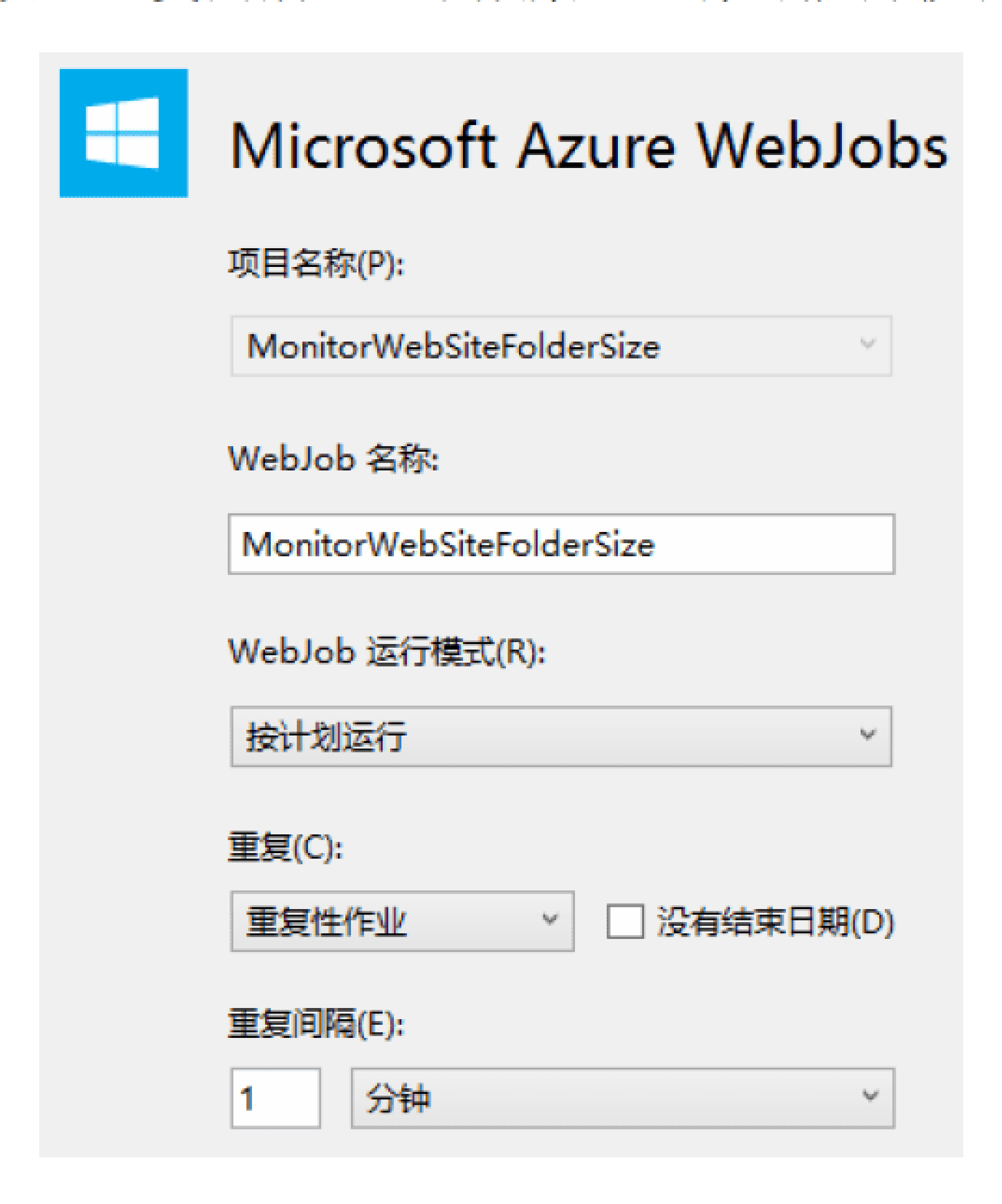


图 7-92　设置 Web 作业运行模式和重复间隔

同样在“添加 Azure WebJob”对话框中，根据需要合理地设置开始与结束时间，如图 7-93 所示。然后单击“确定”按钮。



图 7-93　设置 Web 作业开始与结束时间

如图 7-94 所示，在“发布 Web”对话框中，选择“Microsoft Azure 网站（W）”。登录后，可以选择一个现有网站，或者新建一个网站。在这里，选择一个名为 wzhaotest 的网站。

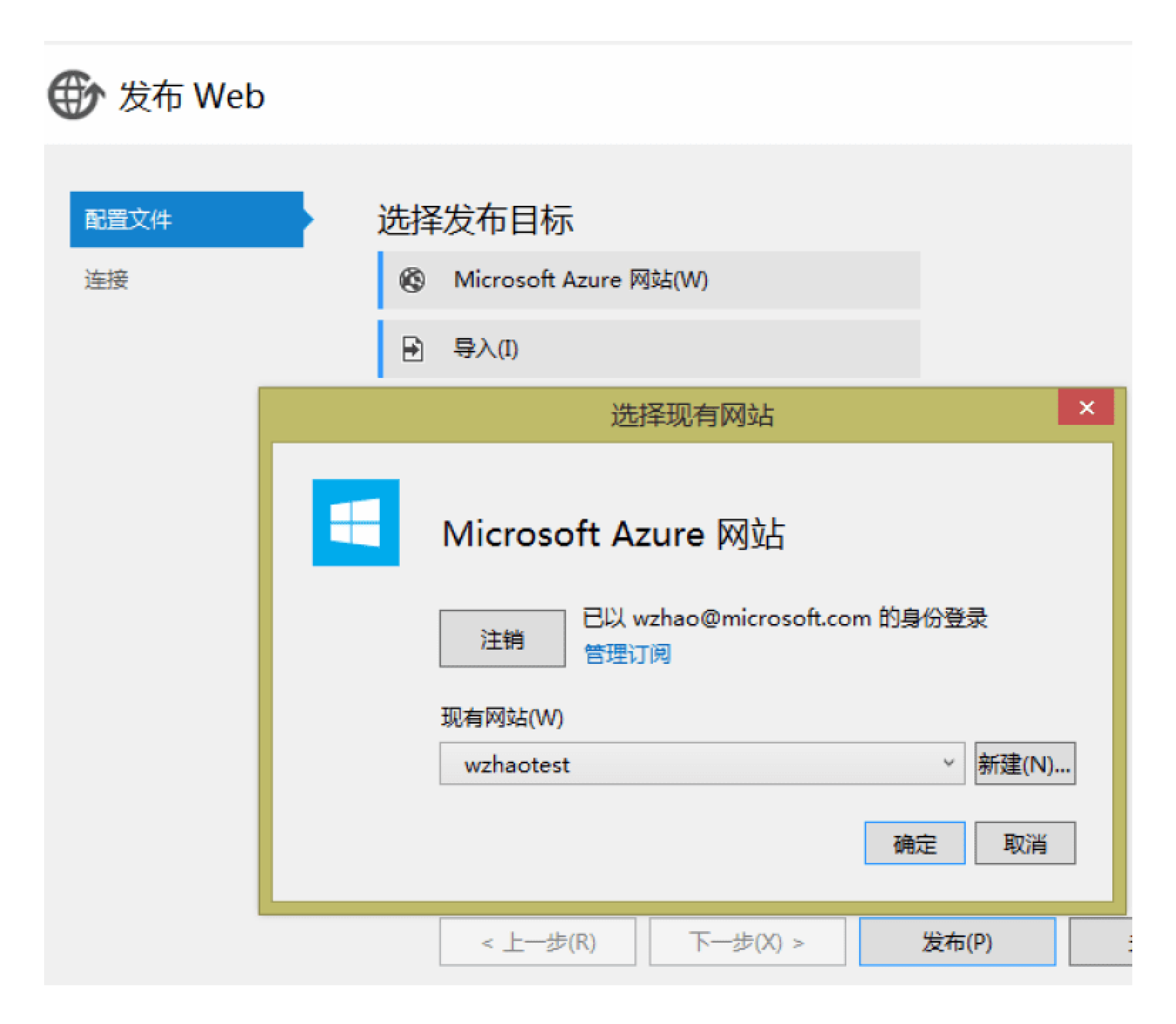


图 7-94　选择发布目标

单击“确定”按钮，然后单击“发布”按钮，将 Web 作业发布到 Azure 网站。

7.9.4.3　检查部署结果

Web 作业发布后，如图 7-95 所示，通过 FTP，可以看到发布后的 Web 作业。

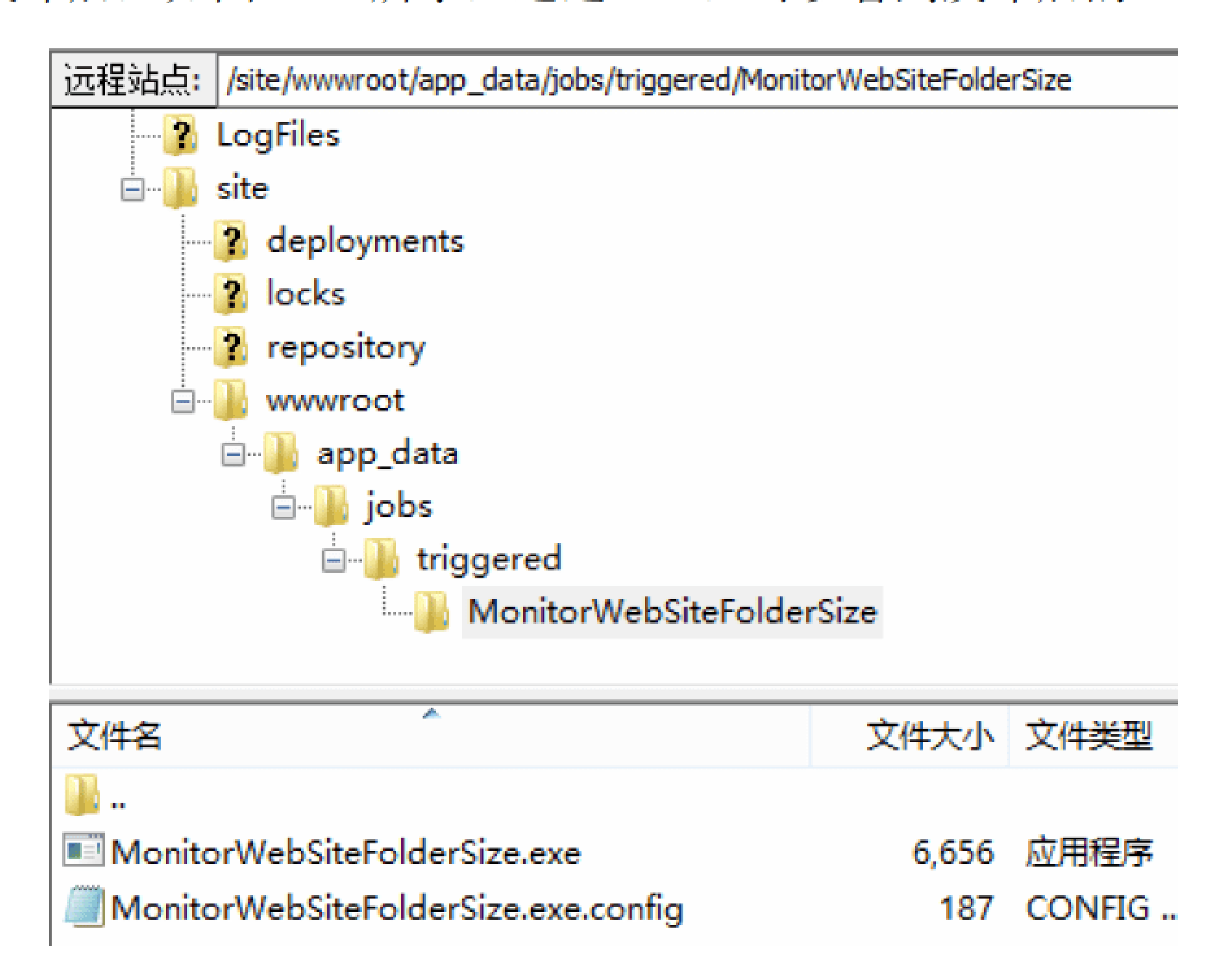


图 7-95　FTP 检查部署后的 Web 作业

如图 7-96 所示，在管理门户网站的“WEB 作业”页面，可以看到作业的运行状态已经显示上一次的运行时间与运行结果。



图 7-96　在管理门户网站中管理 Web 作业

7.9.4.4　查看运行日志

如图 7-97 所示，在“Web 作业管理”页面，单击最右边的日志链接，可以查看 Web 作业的运行日志。

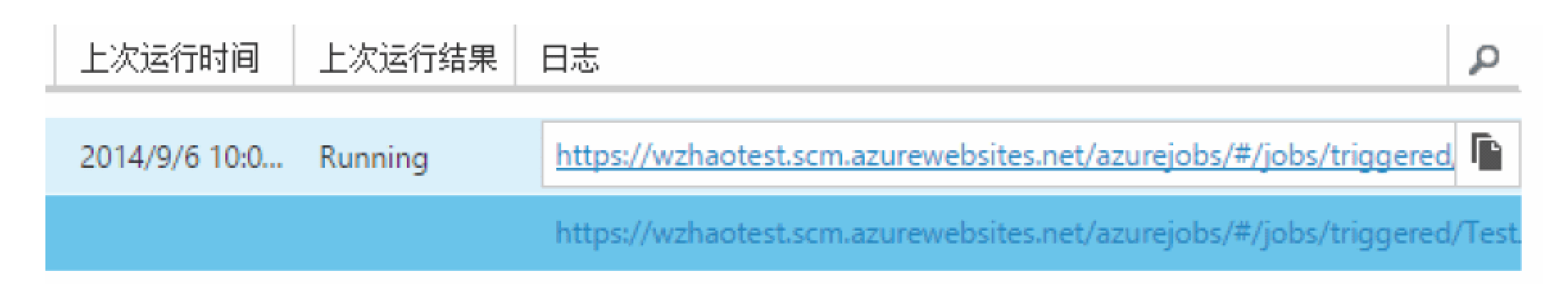


图 7-97　Web 作业日志链接

图 7-98 是 Web 作业执行的日志，单击每次运行的时间，可以查看当时运行的情况。如果应用出现异常，在这里也会看到具体的异常信息，这些异常信息可以协助定位问题。

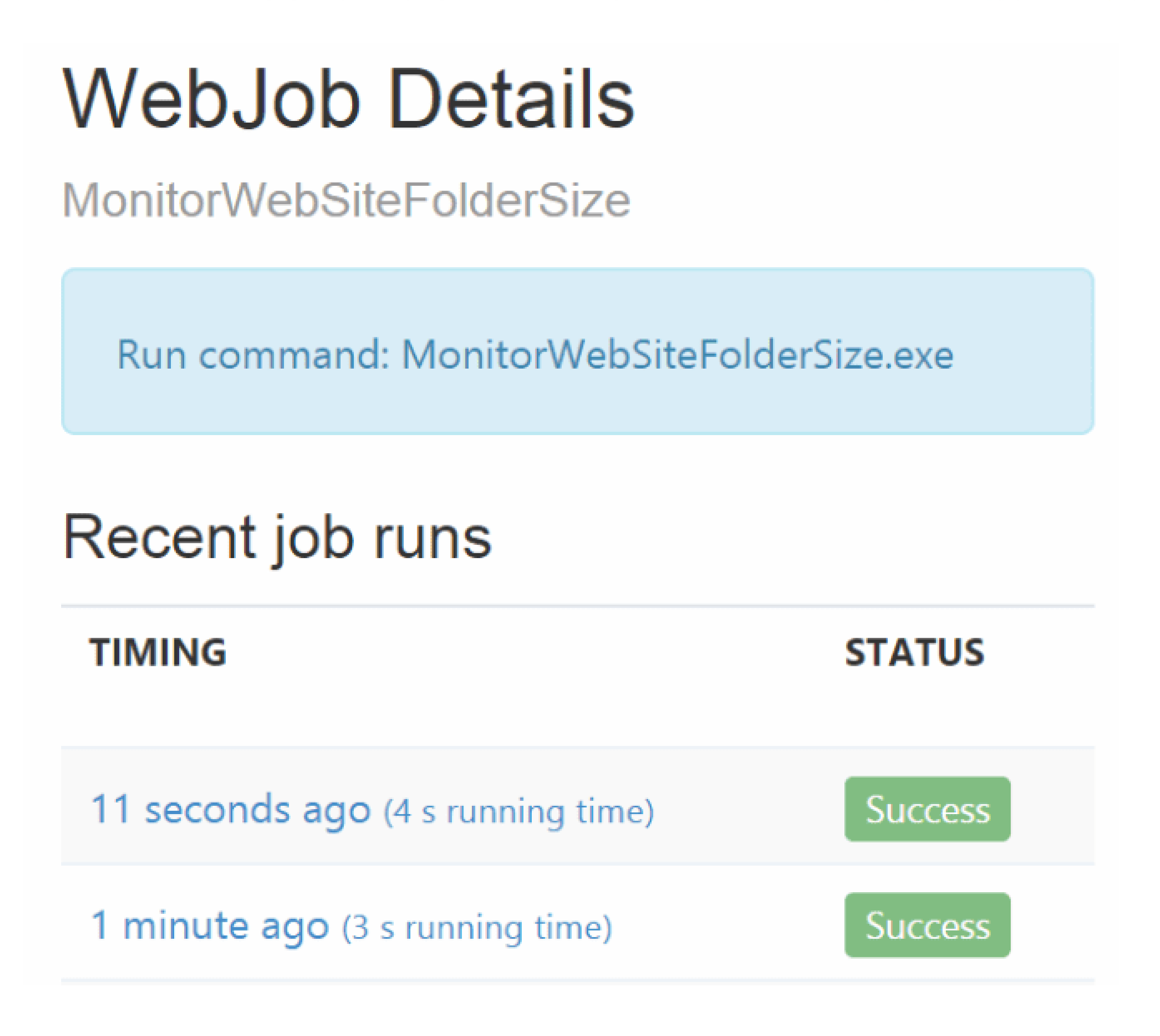


图 7-98　Web 作业执行日志

7.9.4.5　检查运行结果

打开邮箱，可以看到每隔 1 分钟就会收到关于磁盘空间使用情况的邮件，如图 7-99

所示。

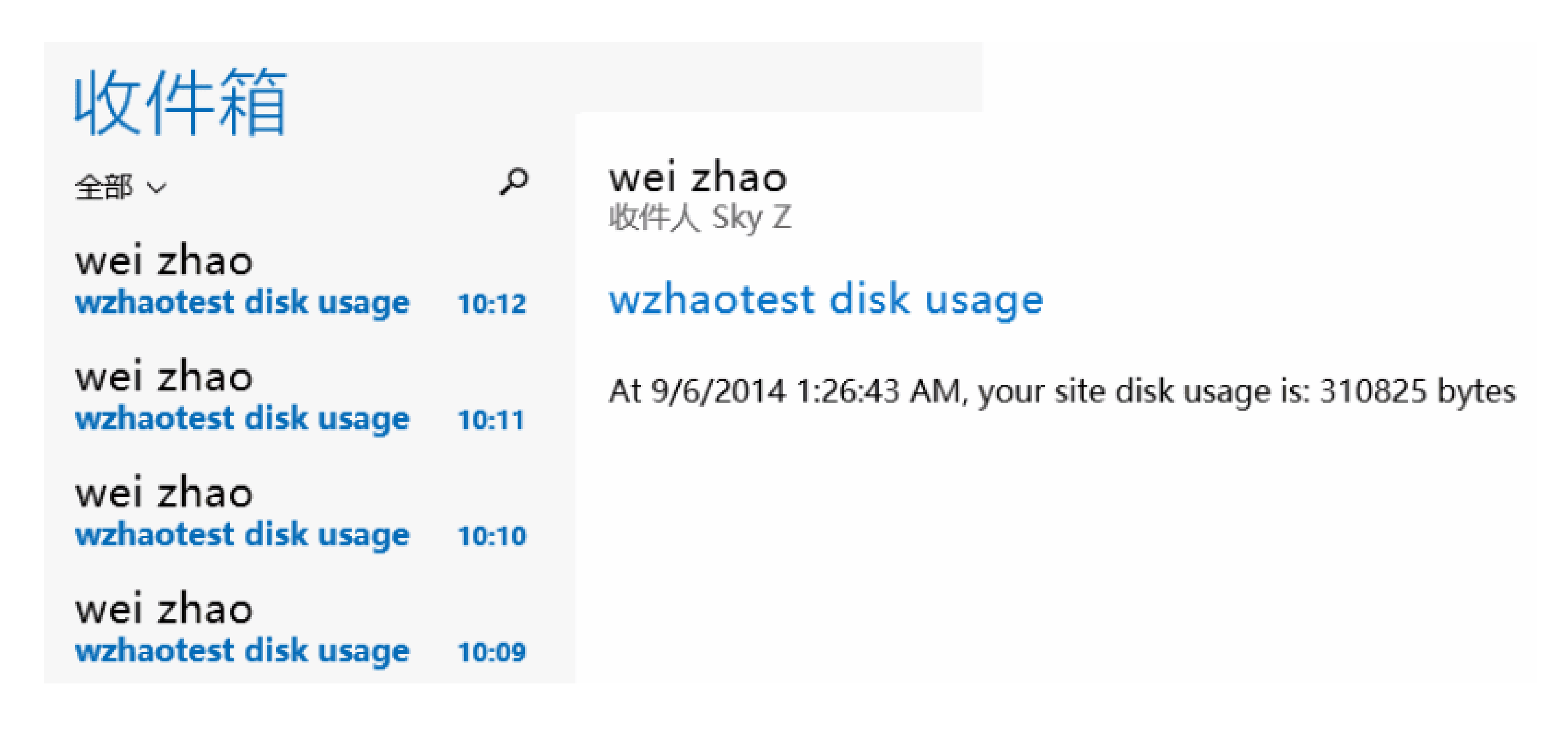


图 7-99　磁盘使用情况邮件

7.9.5　WebJobs SDK

前面演示了如何通过 Web 作业检查磁盘使用情况并发送邮件给管理员。当需要 Web 作业执行复杂任务时，通常需要：

- 连续运行。
- 访问 Azure 存储。

这些需要开发人员编写大量代码。为了简化开发过程，微软公司推出了 WebJobs SDK。通过 WebJobs SDK，开发人员可以专注于应用本身，而不需要编写大量代码实现连续运行或者访问 Azure 存储。

WebJobs SDK 包括以下组件：

（1）NuGet 软件包。开发人员可以通过 NuGet 将 WebJobs SDK 添加到 Visual Studio 项目。该软件包主要包括两个主要组件：

① 绑定与触发系统。WebJobs SDK 支持应用绑定到 Azure 存储 Blob 或者队列，当 Azure 存储有新的内容或新的队列消息时自动触发指定的操作。同时它还支持服务总线（service bus）。

② Web 作业宿主（JobHost）。读取绑定，监听并触发指定的操作。

（2）Web 作业仪表板。这是与 Azure 管理门户网站集成在一起的功能。Web 作业仪表板为基于 WebJobs SDK 的应用提供了丰富的监控和诊断功能。开发人员不需要编写代码即可使用这些监控和诊断功能。

基于 WebJobs SDK 的应用可以运行在任何地方，而不仅仅是运行在 Azure 网站上面。

7.9.5.1　Web 作业宿主

Web 作业宿主支持连续作业和单次运行作业。

1. 连续运行

下面的代码创建一个连续运行的 Web 作业。该作业将会一直运行，应用程序不会退出，

除非被人为终止。当网站被停止时，Web 作业也会被终止。

```
static void Main(string[] args)
{
  JobHost host = new JobHost();
  host.RunAndBlock();
}
```

2. 单次运行

下面的代码创建执行一次的 Web 作业：

```
static void Main(string[] args)
{
    JobHost host = new JobHost();
    host.Call(typeof(Program).GetMethod("MyJobFunction"));
}
```

3. Web 作业宿主配置

Web 作业宿主在启动时会自动读取 Azure 存储的配置。可以在 app.config 文件中通过连接字符串指定。字符串的名称必须为 AzureWebJobsStorage。Azure 存储的账号和访问密钥可以在管理门户获取。

```
<connectionStrings>
    <add name="AzureWebJobsStorage"
        connectionString="DefaultEndpointsProtocol=https;
                         AccountName=<Your Storage Name>;
                         AccountKey=<storage access key>" />
    <add name="AzureWebJobsDashboard" connectionString="…"/>
</connectionStrings>
```

AzureWebJobsDashboard 存储 WebJobs SDK 生成的检测和诊断信息，可以在 Web 作业仪表板中查看这些信息。AzureWebJobsStorage 和 AzureWebJobsDashboard 可以用同一个存储账户或者不同存储账户。

也可以通过 JobHostConfiguration 来指定 Web 作业宿主的配置。JobHostConfiguration 构造函数接收一个连接字符串，该连接字符串会被同时用于 AzureWebJobsStorage 和 AzureWebJobsDashboard。

```
static void Main(string[] args)
{
    JobHostConfiguration config =new JobHostConfiguration("dashboardAnd
    StorageConnectionString ");
    JobHost host = new JobHost(config);
    host.RunAndBlock();
}
```

7.9.5.2 绑定与触发系统

WebJobs SDK 支持绑定到 Azure 存储。绑定后，Web 作业宿主检测下面的事件。当这些事件发生时，Web 作业宿主调用指定的函数。

- 上传了一个新的 Azure 存储 Blob。
- 一个新的 Azure 队列消息。
- 一个新的 Azure 服务总线消息。

1. 新的 Azure 存储 Blob

WebJobs SDK 使用 BlobTrigger 和 Blob 属性绑定到 Azure 存储 Blob。其中，BlobTrigger 是输入，Blob 是输出。下面的函数在 rawimage 容器中有新的 Blob 时被调用。它的功能是对 rawimage 容器中的新 Blob 进行处理，并将处理后内容保存到 process 容器中。

```
public static void BlobToBlob
    ([BlobTrigger("rawimage/{name}")] Stream input,
    [Blob("process/{name} ", FileAccess.Write)] Stream output)
{
    //do some process
    input.CopyTo(output);
}
```

Blob 绑定支持下面 3 种.NET 数据类型，同时支持客户自定义类型。

- Stream。
- TextReader/TextWrite。
- String。

另外，Blob 也支持 Azure SDK 中的存储类型：

- CloudBlob (v1 storage sdk)。
- ICloudBlob。
- CloudPageBlob。
- CloudBlobBlob。

如果应用需要访问 Blob 的一些高级属性，如下面的例子所示，则需要绑定到 Azure SDK 中的类型。

```
public static void
BlobToBlob([BlobTrigger("cloudblobtype/{name}")] CloudBlob input
{
   //Some Advanced Operation
   …
}
```

在使用 Blob 绑定时，需要注意以下事项：

- 在 Web 作业启动时，绑定的容器中的内容会被立即处理。
- WebJobs SDK 通过扫描 Azure 存储日志来检测是否有新的 Blob。Azure 存储日志的

更新周期为 10 分钟。因此，当新的 Blob 上传后，函数可能 10 分钟后才会被触发。如果要求更快的响应，就需要使用 Azure 存储队列。

- 只有当源 Blob 比目标 Blob 更新的时候，才会触发函数调用。

2. 新的队列消息

WebJobs SDK 使用 QueueTrigger 和 Queue 属性绑定到 Azure 存储队列。其中，QueueTrigger 是输入，Queue 是输出。下面的函数在新的消息被加入到 orders 队列时被调用：

```
public static void QueueToBlob([QueueTrigger("orders")] string customerOrders)
{
    //process queue message
    …
}
```

Azure 存储队列绑定支持下面的数据类型：

- CloudQueueMessage。
- String。
- byte[]。
- 用户定义类型（被序列化成 JSON）。

7.9.5.3　WebJobs SDK 实例

下面的文档非常具体地描述了如何利用 WebJobs SDK 开发一个 Web 作业在后台处理客户上传的图片：

Get Started with the Azure WebJobs SDK

http://azure.microsoft.com/en-us/documentation/articles/websites-dotnet-webjobs-sdk-get-started/

在 GitHub 上，可以找到更多的实例代码：

https://github.com/azure/azure-webjobs-sdk-samples

在这里将利用 Web 作业将一个典型的 Web 应用改造成适合全球部署的高性能、高可靠性网站。图 7-100 描述了一个典型的 Web 应用架构。该 Web 应用后台使用数据库，允许客户上传图片到本地存储。

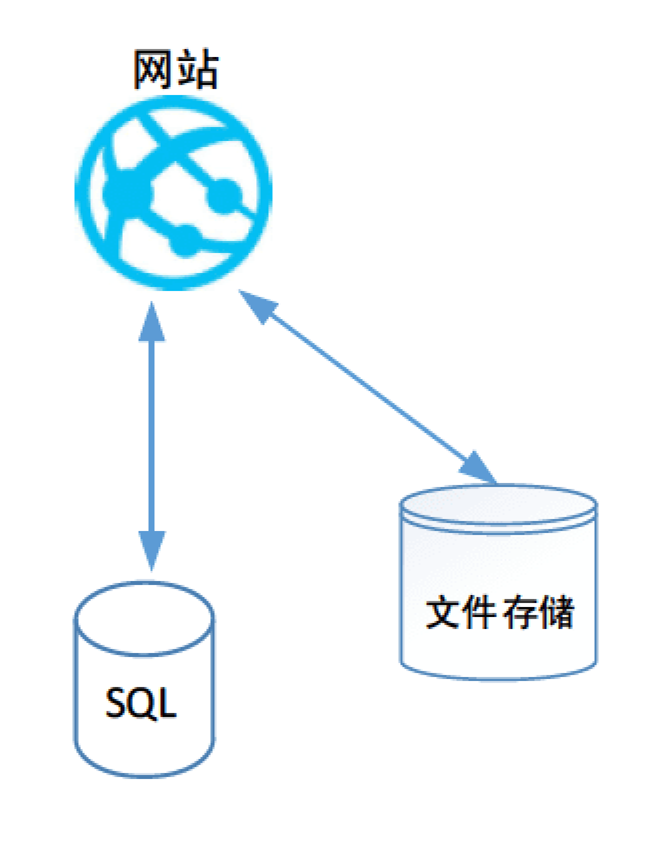


图 7-100　典型 Web 应用架构

该架构中，客户图片文件保存在本地，导致网站不能分布部署到全球数据中心。

图 7-101 描述了改造后的网站架构。在该架构中，将客户上传的文件保存到 Azure 存储中。基于性能的考虑，Azure 存储与网站都位于同一个数据中心。通过 Web 作业来同步两个数据中心的 Azure 存储中客户上传的内容。这样，无论客户的请求被分配到哪个数据中心，都可以保证两个数据中心之间的数据是同步的、一致的。

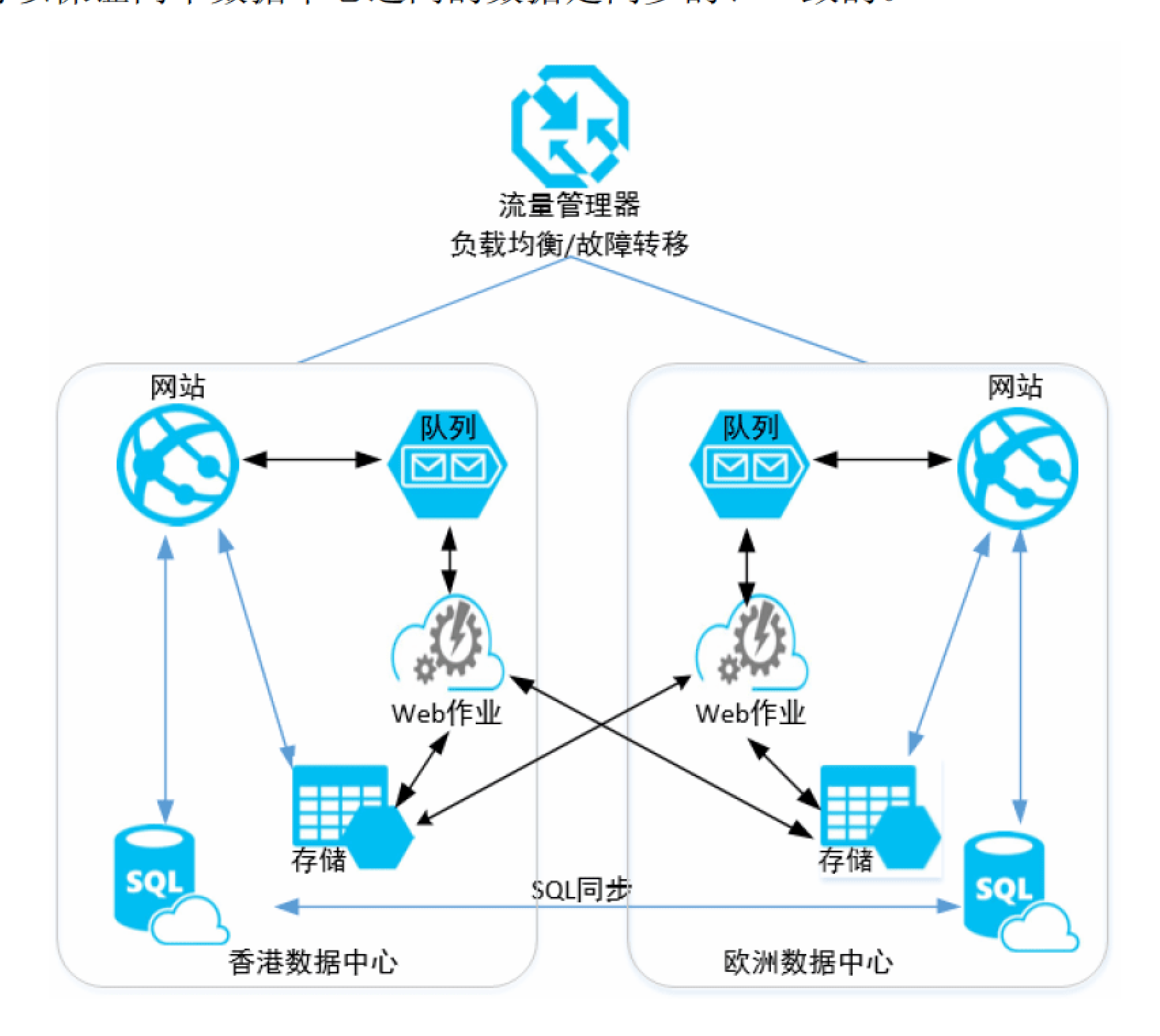


图 7-101　高性能、高可靠性 Azure 网站架构

在本节中详细介绍该架构中 Web 作业的具体实现。

1. 上传图片到 Azure 存储

首先，修改网站的文件上传代码。如下所示，将客户上传的内容图片保存到 Azure 存储中，并且向 Azure 存储队列中发送一条消息。

```
protected void BtnUpload Click(object sender, EventArgs e)
{
   if (FileUploadCtrl.HasFile)
   {
      this.SaveImage(
         Guid.NewGuid().ToString(),
         FileUploadCtrl.FileName,
         FileUploadCtrl.PostedFile.ContentType,
         FileUploadCtrl.FileBytes
      );
      BlobInfo blobInfo = new BlobInfo() {
```

```
                BlobName = FileUploadCtrl.FileName,
                IsSync = false };
            var queueMessage = new CloudQueueMessage(
                    JsonConvert.SerializeObject(blobInfo));
            Global.thumbnailRequestQueue.AddMessage(queueMessage);
            ListBlobsInContainer();
        }
    }
```

2. 图片处理与同步

在 Web 作业中，首先绑定到 Azure 存储队列。当有客户文件上传时，该方法被调用。该方法首先生成缩略图（thumbnail），然后将原始文件复制到其他数据中心的 Azure 存储，并发送一条消息到远端的 Azure 存储队列。远端的 Azure 存储队列收到消息后再次生成缩略图。这样就保证了两个数据中心的文件始终是同步的。

```
    public static void QueueToBlob(
        [QueueTrigger("thumbnailrequest")] BlobInfo blob,
        [Blob("rawimage/{BlobName}")] Stream input,
        [Blob("thumbnail/{BlobName}", FileAccess.Write)] Stream output)
    {
        GenerateThumbNail(input, output);
        //uploaded by customer, sync to remote
        if (!blob.IsSync)
        {
            var targetBlob = remoteContainer.GetBlockBlobReference(blob.BlobName);
            targetBlob.StartCopyFromBlob(
                localContainer.GetBlockBlobReference(blob.BlobName));
            blob.IsSync = true;
            var queueMessage = new CloudQueueMessage(
                    JsonConvert.SerializeObject(blob));
            RemoteThumbnailRequestQueue.AddMessage(queueMessage);
        }
    }
```

完整的源代码和项目工程可以在 www.waws.cn 下载。

7.9.5.4　WebJobs SDK 仪表板

前面提到 WebJobs SDK 提供了仪表板功能以查看和监视运行情况。下面演示如何使用 WebJobs SDK 仪表板。

1. 配置连接字符串

如前所示，使用 WebJobs SDK，必须提供两个 Azure 存储连接字符串。其中一个用于 WebJobs 存储应用数据，另外一个用于仪表板数据存储。要使用 WebJobs SDK 仪表板，需要在管理门户网站配置相同的连接字符串。

首先登录到 Azure 管理门户网站，打开网站的配置页面，在“连接字符串”一栏输入 AzureWebJobsDashbaord 连接字符串，如图 7-102 所示。

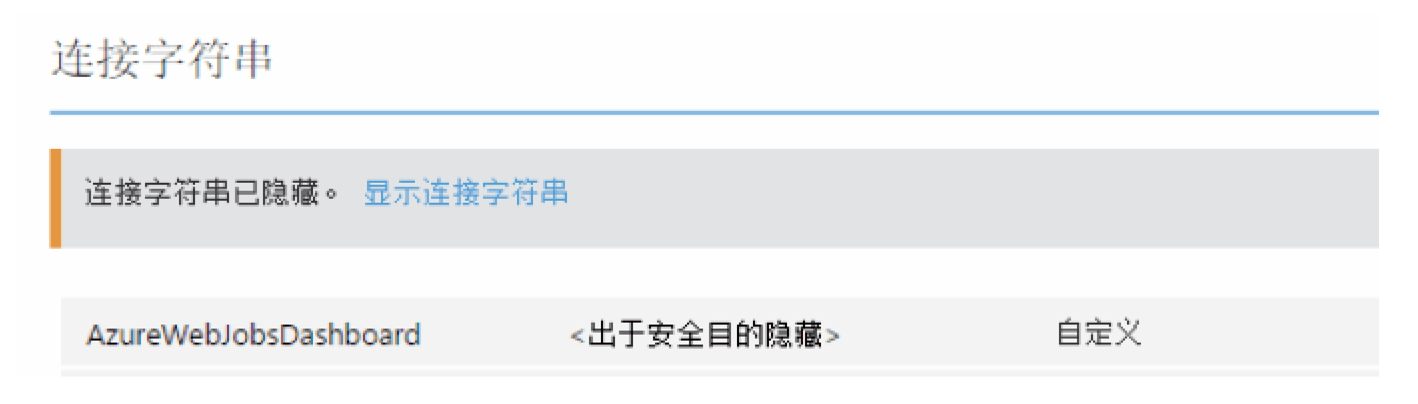


图 7-102 WebJobs SDK 仪表盘连接字符串

2. 打开仪表板

WebJobs SDK 仪表盘连接字符串配置好后，单击页面顶部的“WEB 作业”，然后单击最右边的日志链接（图 7-103），进入仪表板。



图 7-103 Web 作业日志链接

3. 使用仪表板

如图 7-104 所示，可以查看 Web 作业的运行情况，单击 FUNCTION 链接可以查看具体的运行结果和错误信息。

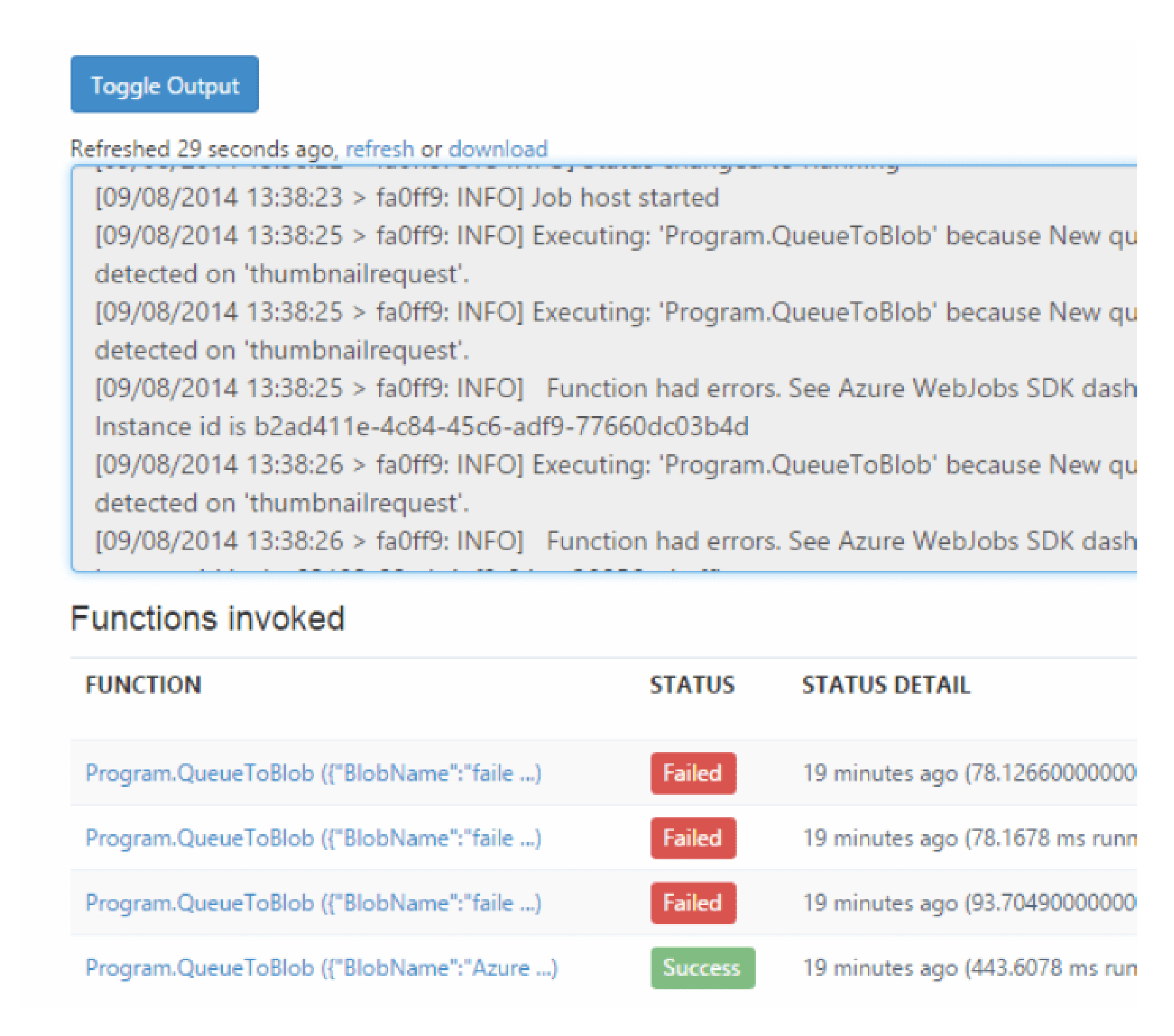


图 7-104 WebJobs SDK 仪表板

7.10　利用 Application Insights 实时洞察用户行为

Application Insights 是微软公司的应用分析解决方案。利用 Application Insights，可以获取用户访问网站的基本信息，比如网站拥有的用户数量以及他们使用该网站的频率。还可以对用户使用特定功能、特定网页，实现特定目标或出现特定错误的频率进行计数，例如查明一局游戏通常持续多长时间。通过这些信息，可以了解用户的使用习惯。开发人员可以专注于最有用的开发工作，改善用户访问体验。

另外，利用 Application Insights，可以确保 Web 服务可用并且能够及时响应。对于 Web 应用，需要了解网站是否出现故障或响应速度缓慢。通过 Application Insights，可以监视网站和 Web 服务的可用性。Application Insights 从遍布全球的多个测试位置向网站发送 Web 请求来模拟实际用户，根据响应结果和响应时间判断网站是否处于可用状态。Application Insights 还可以监视 Web 服务的性能情况，包括网站的资源使用情况以及在不同负载下的响应能力。

在本节中，以一个 Drupal 网站为例，演示如何快速地将 Application Insights 集成到 Azure 网站中，并通过 Azure 管理门户网站查看报告。

7.10.1　获取植入代码

与其他分析软件相同，Application Insights 只需要在网页中植入一段 Java Script 代码。当这些含有 Application Insights 代码的网页被访问时，这段代码会发送一条数据给 Application Insights 服务器，这些数据包括浏览器类型、版本、当前访问的页面等。开发人员可以登录 Azure 管理门户网站获取对应的代码。

（1）登录到测试版本的新版 Azure 管理门户网站 portal.azure.com。

（2）在左侧命令栏，单击“浏览”，选择要集成 Application Insights 的网站，在本例中是 ContosoDrupal 网站。

（3）如图 7-105 所示，单击网站的“分析”区域获取分析代码。

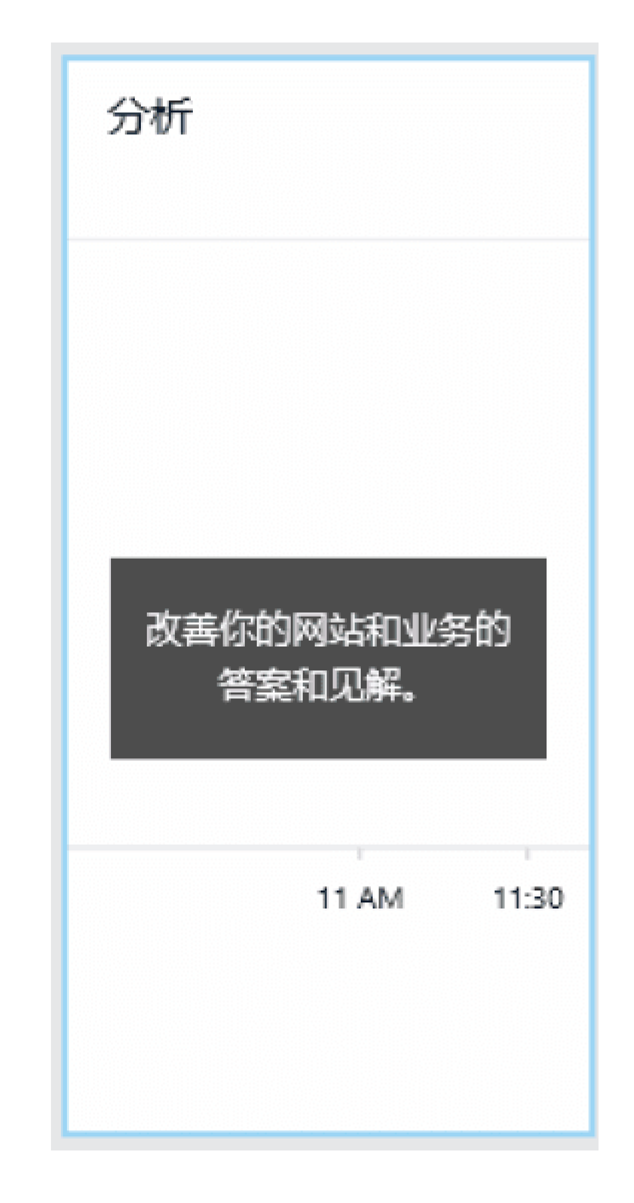


图 7-105　获取网站最终用户使用情况的分析代码

（4）如图 7-106 所示，在门户网站显示最终用户使用情况分析代码。所有网站的植入代码都是相同的，只是每个网站的 instrumentationKey 是不同的。

```
<script type="text/javascript">
    var appInsights=window.appInsights||function(config){
        function s(config){t[config]=function(){var i=arguments;t
.parentNode.appendChild(o),t.cookie=r.cookie,t.queue=[],i=["Event
o),s}),t
    }({
        instrumentationKey:"f948e688-41db-4c1e-9140-6ff715213c88"
    });

    window.appInsights=appInsights;
    appInsights.trackPageView();
</script>
```

图 7-106　AppInsights 代码

7.10.2　植入代码

7.10.2.1　直接修改代码

Application Insights 推荐将代码植入到页面的头部信息，即<head>区域内。通常，基于性能考虑，可以将代码植入到页面结束部分。在 Drupal 中，可以将代码加入到模板文件，这样，Application Insights 的代码会自动加入到 Drupal 的每个页面中。

如图 7-107 所示，将代码加入到 Drupal 模板文件 page.tpl.php 中的 footer 后面。

```
<?php if ($page['footer']): ?>
  <div id="footer" class="clearfix">
    <?php print render($page['footer']); ?>
  </div> <!-- /#footer -->
<?php endif; ?>
<script type="text/javascript">
var appInsights=window.appInsights||function(config){
    function s(config){t[config]=function(){var i=arguments;t.queue.pus
{config:config},r=document,f=window,e="script",o=r.createElement(e),i,u
agName(e)[0].parentNode.appendChild(o),t.cookie=r.cookie,t.queue=[],
config.disableExceptionTracking||(i="onerror",s("_"+i),u=f[i],f[i]=function(
}({
    instrumentationKey:"f948e688-41db-4c1e-9140-6ff715213c88"
});

window.appInsights=appInsights;
appInsights.trackPageView();
</script>
```

图 7-107　在 Drupal 中植入代码

Drupal 的模板文件 page.tpl.php 位于/themes/*ThemeName*/templates 目录下。可以通过 FTP 下载到本地编辑该文件后上传，或者通过 4.7 节介绍的 Visutal Studio 在线编辑功能进行在线编辑。

植入 Application Insights 代码后，访问网站页面时，在页面的 HTML 代码中可以看到 Application Insights 的代码已经被加入到页面中，如图 7-108 所示。

```
  <div class="content">
    <span>Powered by <a href="http://drupal.org">Drupal</a></span>
</div>
  </div>
      </div> <!-- /#footer -->
        <script type="text/javascript">
    var appInsights=window.appInsights||function(config){
        function s(config){t[config]=function(){var i=arguments;t.c
    }({
        instrumentationKey:"f948e688-41db-4c1e-9140-6ff715213c88"
    });

    window.appInsights=appInsights;
    appInsights.trackPageView();
</script>
```

图 7-108　在浏览器中查看植入后的代码

7.10.2.2　通过 URLRewrite 植入代码

对于像 Drupal、WordPress 等内容管理网站，通常所有的页面都基于模板文件。将 Application Insights 代码植入模板文件后，代码自动被加入到所有的页面。这样开发人员可以轻松加入 Application Insights 功能，而无须逐个修改页面文件。

然而，并不是每个客户都了解 Drupal、WordPress 等。此外，大多数客户自己开发的 Web 应用中，每个页面都是独立的，没有模板文件的概念。当植入 Application Insights 代码时，需要编辑每一个需要跟踪的页面，这是一个很大的工作量。在本例中，将演示如何通过 URLRewrite 自动将代码植入到每个页面中。

URLRewrite 的出站规则（outbound rule）用于修改 HTTP 响应。例如，如果网站的导航结构发生了变化，可以创建出站规则，修改 URL 中的内容，以便 Web 页面内容指向正确的位置。在本例中，使用 URLRewrite 的出站规则在 HTTP 响应的</body>之前加入 Application Insights 的代码。

1. 将 Application Insights 代码保存到.js 文件

首先，将 Application Insights 的代码保存到.js 文件中，比如 appInsights.js 文件。注意，文件内容只需要包含图 7-106 所示代码中<script>节点内的内容，无须包含<script>节点。下面是一个示例。

```
var appInsights=window.appInsights||function(config){
function ……
window.appInsights=appInsights;
appInsights.trackPageView();
```

2. 将该文件上传到 Azure 网站

可以使用 FTP、Git 或其他方式上传该文件。也可以利用 4.7 节介绍的 Visual Studio 在线编辑功能在线生成 appInsights.js 文件。在本例中，将该文件上传至网站根目录下的/script 目录。

3. 配置 URLRewrite 出站规则

在网站文件中通常包含很多类型的文件，比如.css（级联样式表）、.js（脚本文

件）、.png/.jpeg（图片文件）等，这些文件无须植入代码。因此，在本例中，通过 isHTML 的前提条件来判断响应是否是 text/html，如果是，则将 Application Insights 的代码植入到</body>之前。

下面创建一个出站规则，在 text/html 响应中查找</body>，如果找到，将其替换为下面的内容。这相当于将 Application Insights 脚本代码植入到</body>结束之前。

```
<script type="text/javascript" src="/script/appinsights.js"></script></body>
```

这样，Application Insights 的脚本被自动加入到每个页面中。需要将以下规则合并到网站的 web.config 文件中：

```
<system.webServer>
  <rewrite>
    <outboundRules>
      <rule name="AppInsight" preCondition="isHTML" patternSyntax="ExactMatch">
        <match filterByTags="None" pattern="&lt;/body>" />
        <action type="Rewrite" value="&lt;script type=&quot;text/javascript&quot;
                 src=&quot;/script/appinsights.js&quot;>&lt;/script>&lt;/body>" />
      </rule>
      <preConditions>
        <preCondition name="isHTML">
          <add input="{RESPONSE_CONTENT_TYPE}" pattern="^text/html" />
        </preCondition>
      </preConditions>
    </outboundRules>
  </rewrite>
    </system.webServer>
```

4. 验证修改结果

再次访问网站，然后在浏览器中查看 HTML 的源码，可以看到下面的内容。这表示 Application Insights 的代码已经被成功植入。

```
<script type="text/javascript" src="/script/appinsights.js"></script></body>
 </html>
```

5. 注意事项

如果页面响应被压缩，URLRewrite 的出站规则就会报错，通常客户端会收到 HTTP 500.52 错误。这是因为当页面响应被压缩后，响应内容变成二进制的字节流而不是文本。URLRewrite 无法在压缩后的结果中查找指定的文本内容。因此，采用该解决方案，需要关闭静态和动态内容压缩功能。

下面的配置可以关闭静态和动态内容压缩功能，需要将该规则合并到网站的 web.config 文件中。

```
<system.webServer>
```

```
    <urlCompression doStaticCompression="false" doDynamicCompression="false" />
    </system.webServer>
```

7.10.3　查看分析报告

新版的 Azure 管理门户网站集成了网站的 Application Insights 的报告查看功能，可以非常方便地通过新的管理门户网站查看分析报告。

（1）登录到测试版本的新版 Azure 管理门户网站 portal.azure.com。

（2）在左侧命令栏，单击“浏览”，选择要集成 Application Insights 的网站，在本例中是 ContosoDrupal 网站。

（3）如图 7-109 所示，现在，网站的分析区域显示分析结果的汇总。

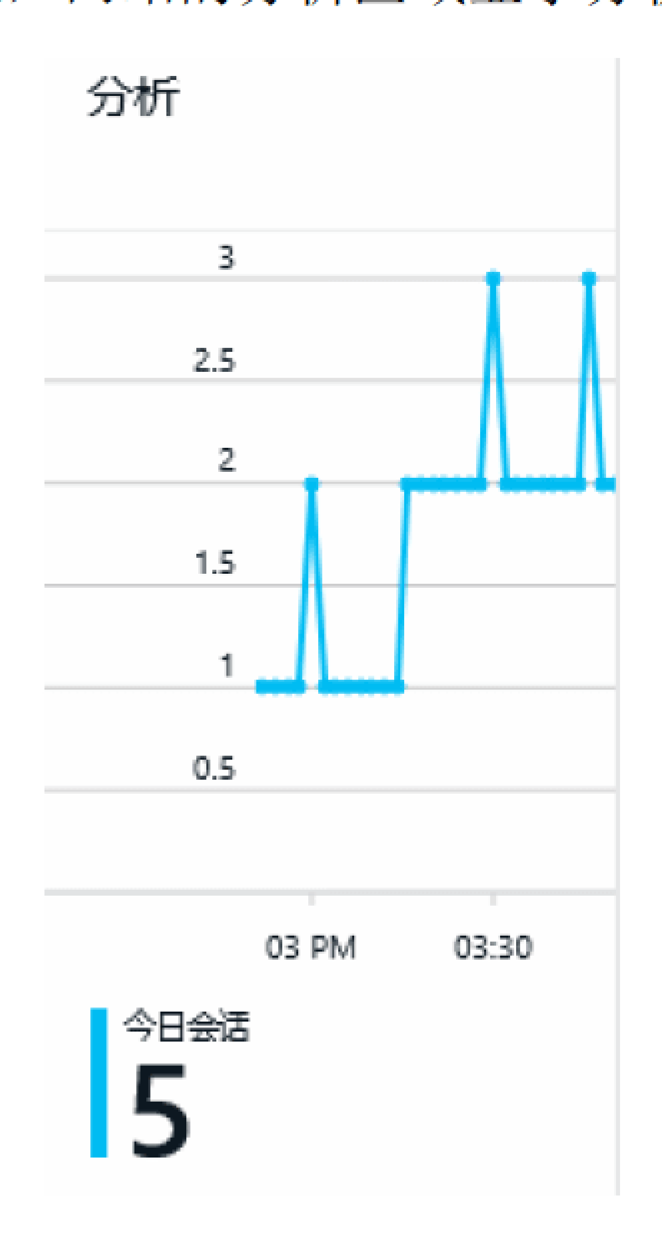


图 7-109　分析结果概览

（4）单击“分析”区域，进入“分析结果”页面，其中包含“环境”、“活动”和“配置”。

（5）如图 7-110 所示，“环境”部分包含了浏览器的分析结果。

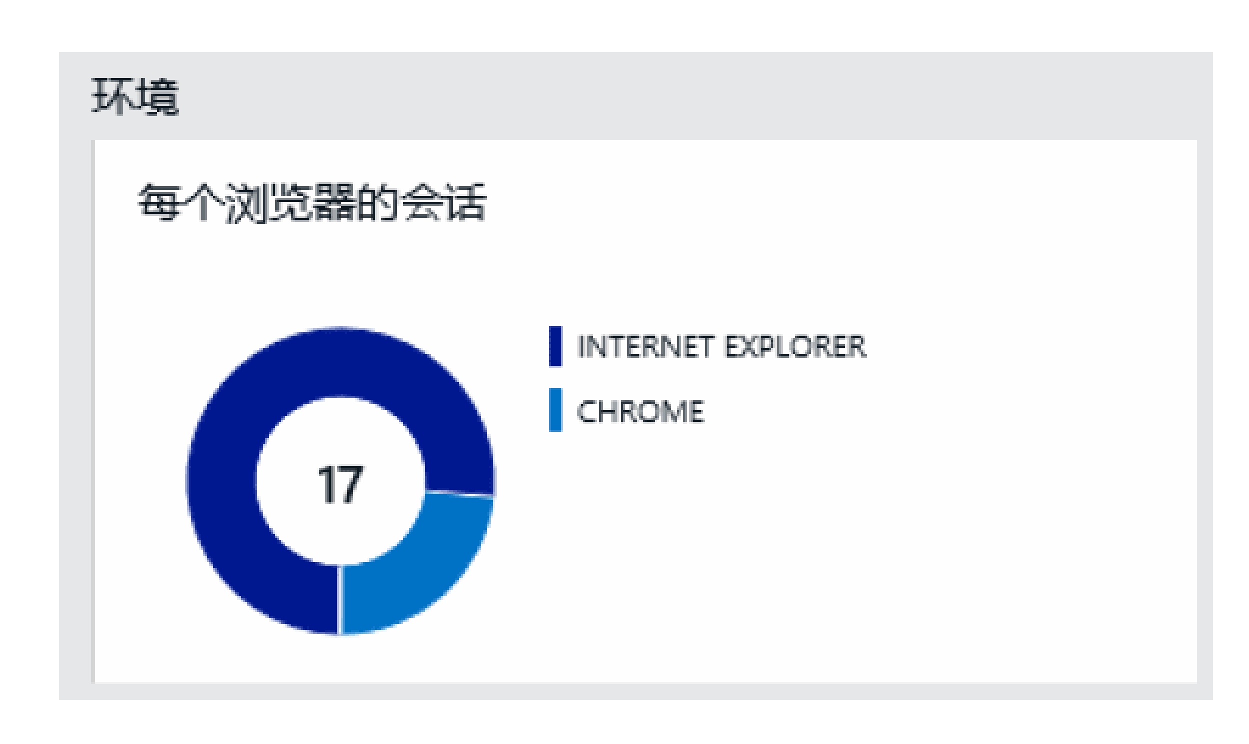


图 7-110　浏览器分析结果

（6）如图 7-111 所示，页面视图包含页面的访问情况分析。包含前五位的页面访问量、总的访问量和最慢的页面等。

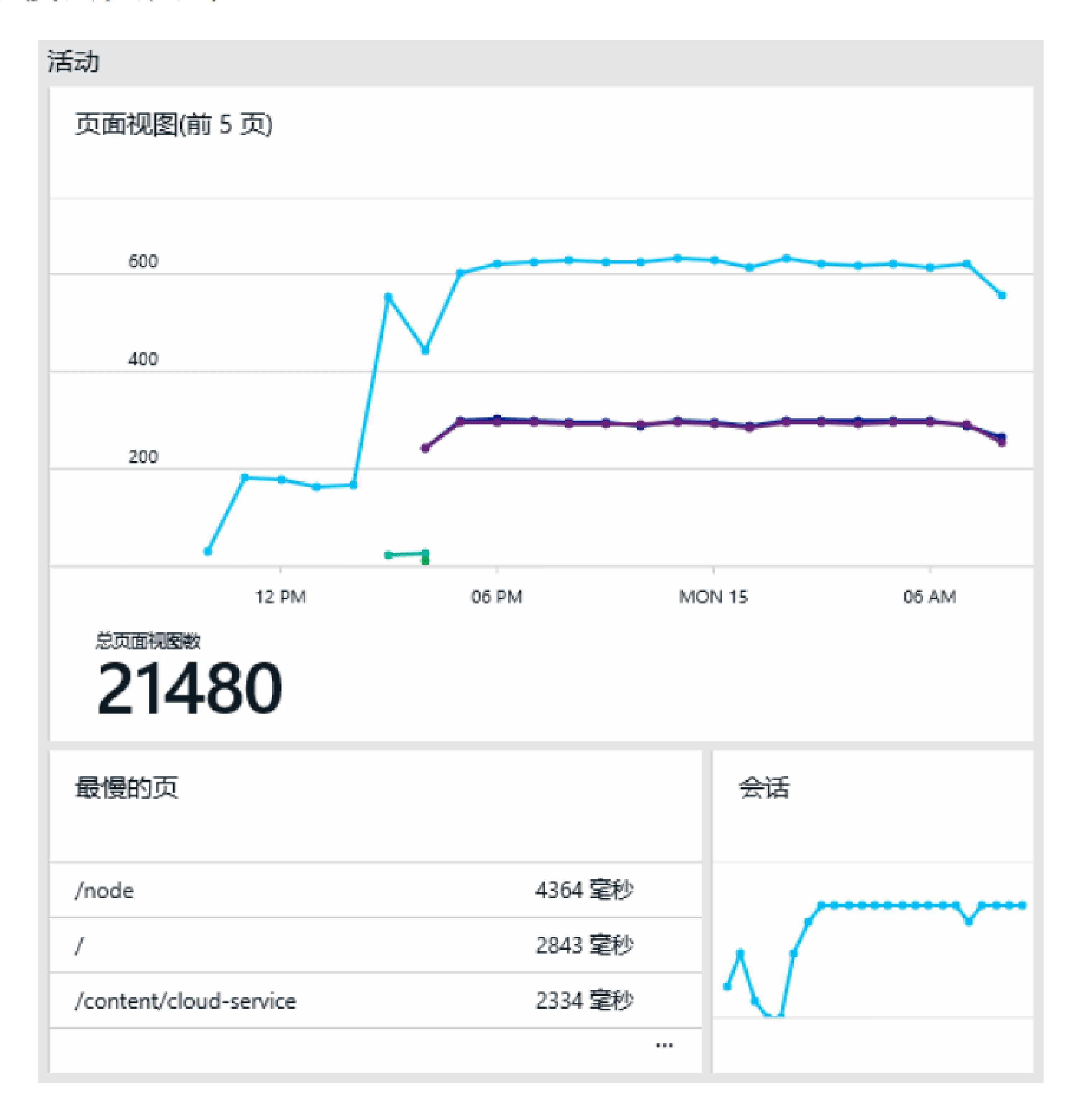


图 7-111 Application Insights 页面视图

7.11 参考文献与扩展阅读

1. Adding Sign-On to Your Web Application Using Azure AD

该文档介绍了使用 Azure 活动目录在 ASP.NET 应用程序中实现单点登录。通过该文档，可以了解 Azure 活动目录的运作方式以及如何将 Azure 活动目录集成到应用中。

http://msdn.microsoft.com/library/azure/dn151790.aspx

2. Create a Line-of-Business Application on Azure Web Sites

Azure 网站官方文档，介绍了基于 Azure 网站创建关键企业级应用。

http://azure.microsoft.com/en-us/documentation/articles/web-sites-business-application-solution-overview/

3. Connect an Azure web site to an on-premises resource using Hybrid Connections

Azure 网站官方文档，介绍了如何利用混合连接将 Azure 网站与企业私有数据中心集

成，实现混合云解决方案。

http://azure.microsoft.com/en-us/documentation/articles/web-sites-hybrid-connection-get-started/

4. Getting Started with ASP.NET Web API 2 (C#)

ASP.NET 官方文档，介绍了如何利用 APS.NET Web API 2 创建 HTTP Web API。

http://www.asp.net/web-api/overview/getting-started-with-aspnet-web-api/tutorial-your-first-web-api

5. Traffic Manager Overview

Azure 官方文档，介绍了流量管理器的概念，以及如何配置流量管理器。

http://msdn.microsoft.com/en-us/library/azure/hh744833.aspx

6. How to use CDN—Azure feature guide

Azure CDN 提供了高带宽全球内容交付的解决方案。利用遍布全球的 CDN 节点可以提高关键应用的性能。该文章介绍了如何创建和使用 Azure CDN。

http://azure.microsoft.com/en-us/documentation/articles/cdn-how-to-use/

7. Hybrid Connections Overview

混合连接提供一个简单的连接 Azure 网站和企业私有数据中心的解决方案。该文章介绍了混合连接的基本概念和应用场景。

http://azure.microsoft.com/en-us/documentation/articles/integration-hybrid-connection-overview/

8. Azure Managed Cache Service

利用 Azure 缓存服务可以构建高速、可扩展的 Web 应用，尤其是在突发负载的时候，可以提高系统的响应能力。该文章介绍了 Azure 缓存服务的特点以及如何计划、创建、开发、管理和使用 Azure 缓存服务。

http://msdn.microsoft.com/en-us/library/azure/dn386094.aspx

9. Get Started with the Azure WebJobs SDK

WebJobs SDK 提供了访问 Azure 存储的简单解决方案。该文档介绍了如何在 Azure 网站中利用 Azure WebJobs SDK 开发后台应用访问 Azure 存储。

http://azure.microsoft.com/en-us/documentation/articles/websites-dotnet-webjobs-sdk-get-started/

10. Webjobs SDK

Webjobs SDK 是一个开发框架，通过 Webjobs SDK 可以轻松创建访问 Azure 存储的后台应用。

http://blogs.msdn.com/b/jmstall/archive/2014/01/24/webjobs-sdk.aspx

11. Scaling Your Web Application with Azure Web Sites

该文档介绍了如何通过集成其他 Azure 服务创建高性能、可扩展的 Azure 网站应用。

http://msdn.microsoft.com/en-us/magazine/dn786914.aspx

12. Architect for the Cloud Using Azure Web Sites

该文档介绍了高性能、可扩展的 Azure 网站应用架构。

http://msdn.microsoft.com/en-us/magazine/dn787017.aspx

13. Analytics for Microsoft Azure websites

该文档介绍了如何在 Azure 网站中集成 Application Insights。

http://azure.microsoft.com/en-us/documentation/articles/insights-usage-analytics/

14. Adding Analytics to Azure Web Sites

该视频演示了如何在 Azure 网站中集成 Application Insights。

http://azure.microsoft.com/en-us/documentation/videos/adding-analytics-to-azure-web-sites/

15. Creating Outbound Rules for URLRewrite Module

该文档介绍了如何使用 URLRewrite 的出站规则修改 HTTP 响应。

http://www.iis.net/learn/extensions/url-rewrite-module/creating-outbound-rules-for-url-rewrite-module

第 8 章　高 级 专 题

8.1　使用 Kudu 站点

前面介绍 Git 部署方式的时候介绍了 SCM 网站。每一个 Azure 网站都有一个与之对应的 SCM 网站。SCM 网站负责运行 Kudu 和网站扩展。Kudu 网站除了支持网站部署之外，还提供非常强大的功能。本节将具体介绍这些功能。

Kudu 网站的 URL 为 https://sitename.scm.azurewbsites.net，比如网站

http://contoso.azurewebsites.net

对应的 SCM 网站 URL 为

https://contoso.scm.azurewebsites.net/

注意事项：

（1）SCM 网站必须使用 HTTPS 访问。

（2）SCM 网站不受自定义域名的影响，不能通过自定义域名访问。

（3）SCM 网站要求认证，需要使用 Azure 订阅账户登录或者使用部署凭据登录。采用部署凭据登录时采用基本认证（basic authentication），对应的 URL 为

https://contoso.scm.azurewebsites.net/BasicAuth

8.1.1　关于 Kudu 架构

Kudu 采用单租户架构。每个 Azure 网站都有自身对应的 SCM 网站，与其他 Azure 网站完全隔离。SCM 网站本身以 Azure 网站方式运行。

SCM 网站与对应的 Azure 网站运行在同一个安全上下文中。两个网站的 IIS 应用程序池使用相同的用户身份。因此 Azure 网站不能访问的资源，SCM 网站也无法访问。另外，SCM 网站与 Azure 网站共享资源配额限制，比如 CPU、内存、磁盘空间等。

图 8-1 描述了 Kudu 的架构。

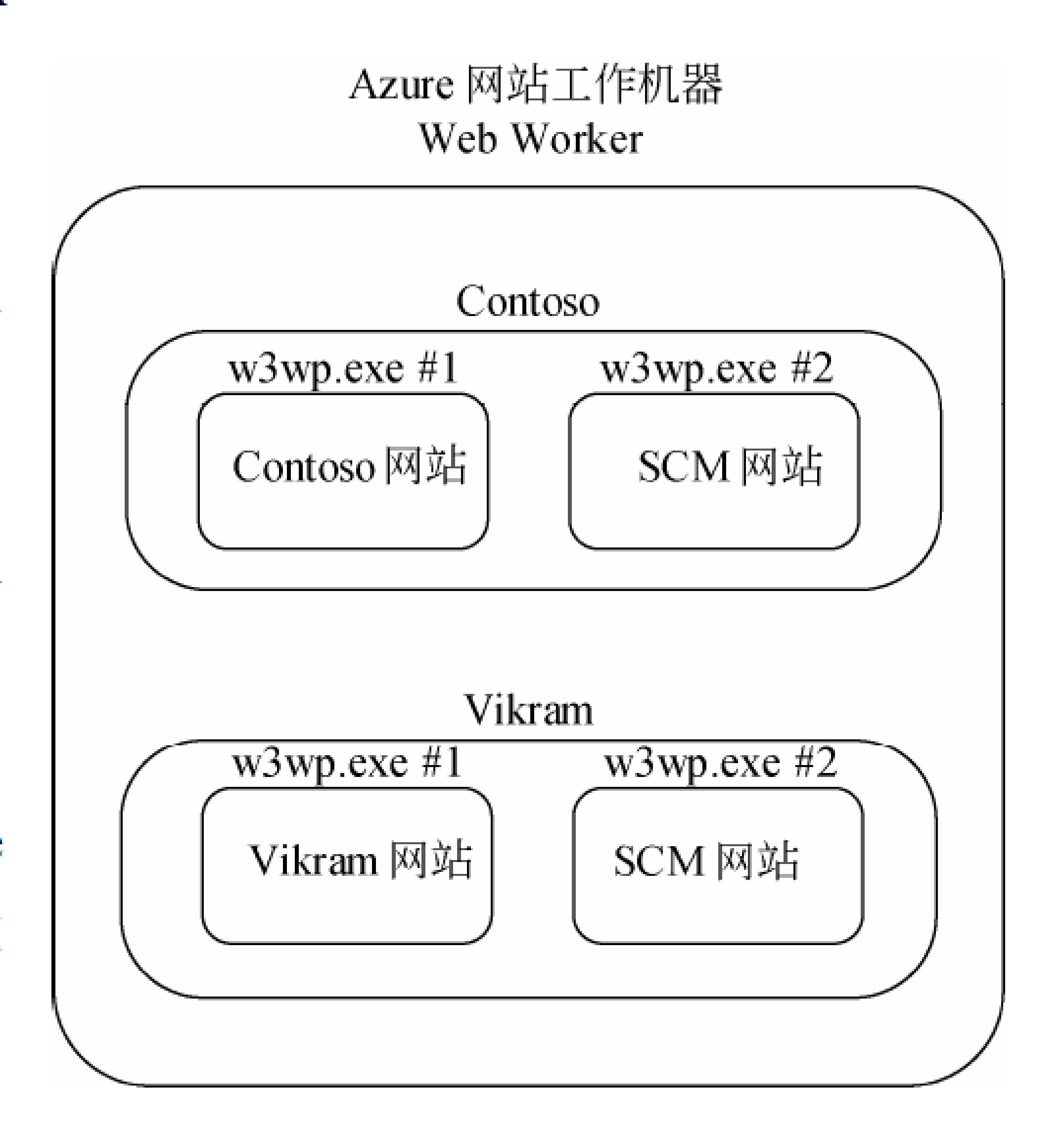


图 8-1　Kudu（SCM）架构

8.1.2　使用 Kudu 控制台

Kudu 控制台是 SCM 网站提供的一个基于 Web 的工具，通过 Kudu 控制台，可以浏览

网站的文件、运行命令、使用网站诊断 REST API 等。下面简单介绍 Kudo 控制台的一些常用的功能。

8.1.2.1　查看运行环境

登录到 SCM 网站后，单击顶部的 Environment 菜单，可以看到所有网站运行时的配置，包括：

- 系统信息（System Info）。
- 应用设置（AppSettings）。
- 连接字符串（Connection Strings）。
- 环境变量（Environment Variables）。
- 路径设置（PATH）。
- HTTP 头部信息（HTTP headers）。
- HTTP 服务器变量（Server variables）。

8.1.2.2　运行命令

Kudu 控制台提供了两种运行命令的方式：Windows 命令行和 PowerShell 控制台。可以通过 Kudu 控制台顶部的 Debug console 菜单选择。

1. Windows 命令行

图 8-2 描述了在控制台运行 dir 命令的结果。

```
Kudu Remote Execution Console
Type 'exit' then hit 'enter' to get a new CMD process.
Type 'cls' to clear the console

Microsoft Windows [Version 6.2.9200]
(c) 2012 Microsoft Corporation. All rights reserved.

D:\home>dir
 Volume in drive D is Windows
 Volume Serial Number is A4D7-FB16

 Directory of D:\home

06/03/2014  12:16 PM    <DIR>          .
06/03/2014  12:16 PM    <DIR>          ..
06/03/2014  12:16 PM    <DIR>          data
03/10/2014  09:09 AM    <DIR>          LogFiles
06/03/2014  12:16 PM    <DIR>          site
               0 File(s)              0 bytes
```

图 8-2　Kudu 命令行控制台

2. PowerShell 控制台

图 8-3 描述了使用 PowerShell 控制台获取当前进程的结果。

```
Kudu Remote Execution Console
Type 'exit' then hit 'enter' to get a new powershell process.
Type 'cls' to clear the console

PS D:\home> get-process

Handles  NPM(K)    PM(K)      WS(K) VM(M)   CPU(s)     Id ProcessName
-------  ------    -----      ----- -----   ------     -- -----------
     56       8     3364       4368    32     0.03  55952 cmd
    160      14    11184      15408   120     2.33   1476 DaaSRunner
    284      23    44672      48708   199     2.33  16488 powershell
   2229     170   213964     255224   810   115.53  10476 w3wp


PS D:\home> █
```

图 8-3　Kudu PowerShell 控制台

8.1.2.3　利用 Windows 命令行抓取内存转储文件

内存转储文件（dump）是一种特殊的二进制文件，它包含了系统或者某个指定的进程在特定时间点的所有驻留在内存中的内容，比如栈空间、堆（heap）以及句柄等。分析内存转储文件是查找应用异常和性能瓶颈的最佳方法。有很多工具可以生成内存转储文件，比如 Windows 调试工具、ProcDump、DebugDiag 等。与运行在企业内部的系统不同，无法通过远程桌面登录到运行网站的虚拟机去抓取内存转储文件。下面介绍如何通过 Kudu 控制台利用 ProcDump 工具抓取 IIS 工作进程的内存转储文件。

（1）将 ProcDump 通过 FTP 上传到网站，比如 wwwroot/dump/目录下。

可以在下面的地址下载 ProcDump：

http://technet.microsoft.com/en-us/sysinternals/dd996900.aspx

（2）获取当前工作进程 ID。

在 Kudu 控制台中，打开 Windows 命令行，运行 powershell.exe -Command Get-Process 命令，如图 8-4 所示，该命令返回当前的 IIS 工作进程信息。

```
Handles  NPM(K)    PM(K)      WS(K) VM(M)   CPU(s)     Id ProcessName
-------  ------    -----      ----- -----   ------     -- -----------
     49       8     3572       5128    33     0.03   9108 cmd
     49       8     3544       5092    32     0.05  16696 cmd
    143      15    11264      15732   125     0.27   3308 DaaSRunner
    339      28    46008      49468   201     2.33  27740 powershell
    789      92    76948      88544   730     4.41  31692 w3wp
```

图 8-4　获取进程 ID

（3）运行 cd site\wwwroot\dump 命令，切换到上传 ProcDump 工具的目录。

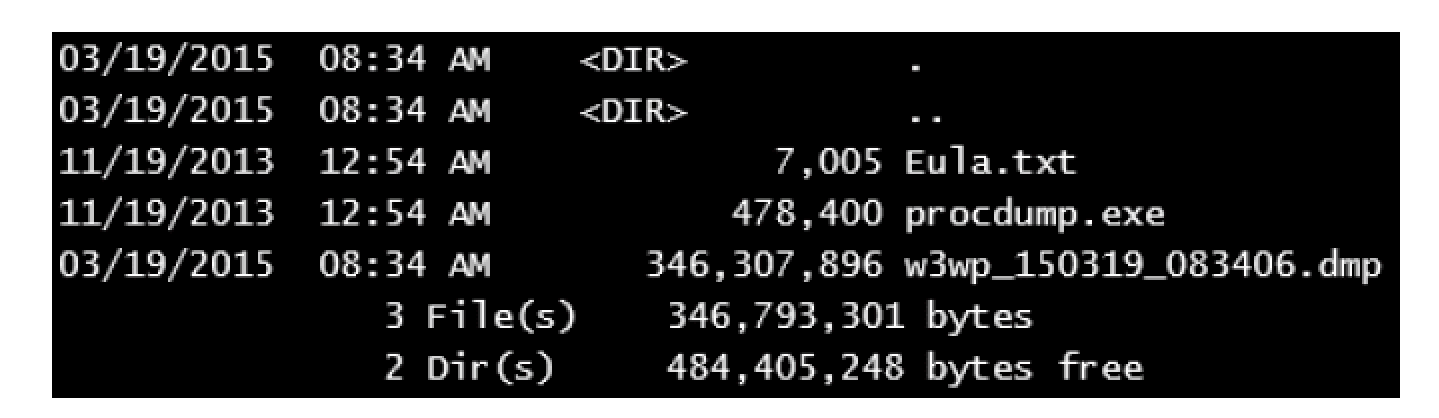
（4）运行 ProcDump -accepteula -ma 28456 命令，如图 8-5 所示，会看到内存转储文件已经生成。

```
ProcDump v6.00 - Writes process dump files
Copyright (C) 2009-2013 Mark Russinovich
Sysinternals - www.sysinternals.com
With contributions from Andrew Richards

Writing dump file D:\home\site\wwwroot\debugger\w3wp_150319_083406.dmp
Writing 337MB. Estimated time (less than) 11 seconds.
Dump written.
```

图 8-5　运行 ProcDump 命令

（5）运行 dir 命令，可以看到内存转储文件的信息，如图 8-6 所示。



```
03/19/2015  08:34 AM    <DIR>          .
03/19/2015  08:34 AM    <DIR>          ..
11/19/2013  12:54 AM             7,005 Eula.txt
11/19/2013  12:54 AM           478,400 procdump.exe
03/19/2015  08:34 AM       346,307,896 w3wp_150319_083406.dmp
               3 File(s)    346,793,301 bytes
               2 Dir(s)    484,405,248 bytes free
```

图 8-6　内存转储文件

注意：内存转储文件的大小与应用使用的内存情况紧密相关，需要注意避免因为生成内存转储文件而导致超出磁盘空间配额的情况。

8.1.3　文件管理

登录到 Kudu 控制台，单击顶部的 Debug console 菜单，选择 Windows 命令行或者 PowerShell。在命令行控制台或者 PowerShell 控制台上方，如图 8-7 所示，显示文件浏览器。

/ + | 3 items

	Name	Modifed
	data	2014年6月3日 20:16:54
	LogFiles	2014年6月3日 20:56:19
	site	2014年6月3日 21:26:22

图 8-7　Kudu 文件管理器

8.1.3.1　文件夹导航

单击任何一个文件夹的名称，可以打开该文件夹。在任意文件夹，单击会回到网站的根目录 d:\home。单击会进入网站的配置文件夹。在配置文件夹，可以查看网站的 applicationHost.config 文件。单击，可以查看系统盘的内容（基于安全考虑，有些文件已经被屏蔽。）。

8.1.3.2　文件操作

如图 8-8 所示，可以通过拖曳的方式将本地文件上传到 Azure 网站。

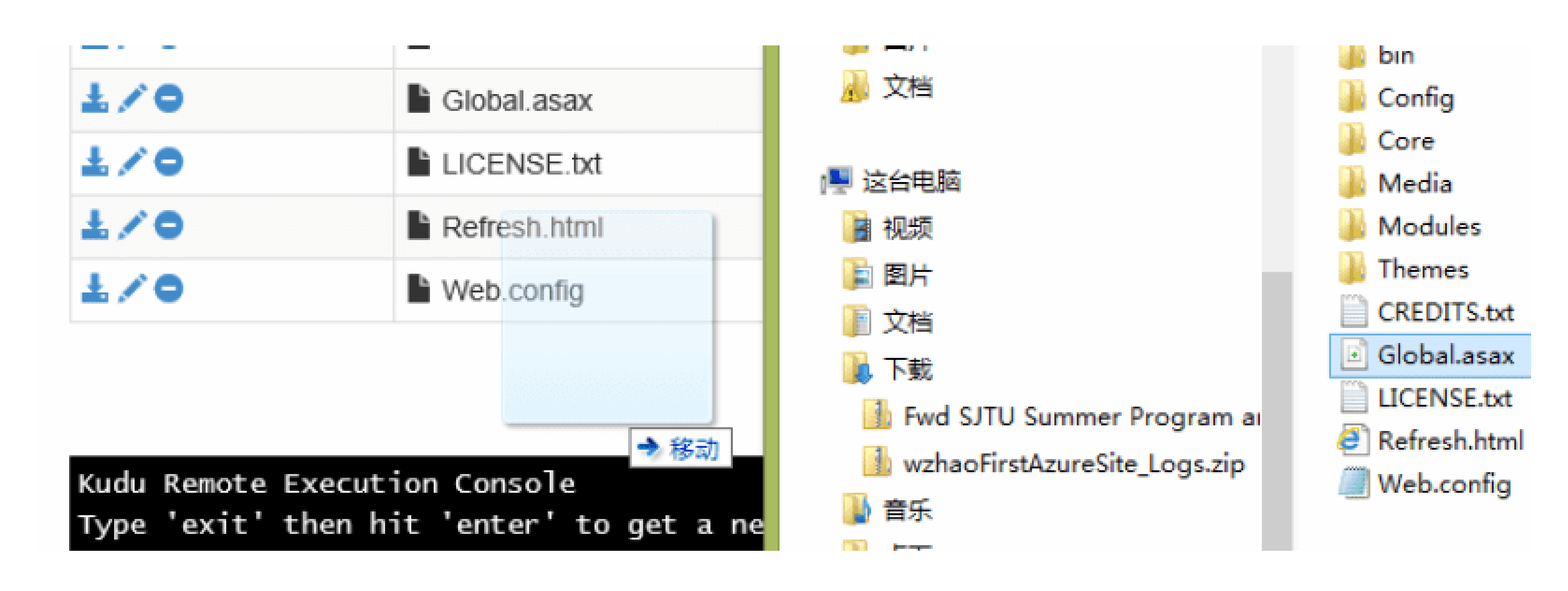


图 8-8　通过拖曳的方式上传文件

文件左边的 3 个图标依次表示下载、在线编辑和删除。

8.1.4　进程管理

登录到 Kudu 控制台，单击顶部的 Process explorer，如图 8-9 所示，可以看到当前网站的进程信息。

Process Explorer

Find Handle...

Refresh

name	pid	total_cpu_time	working_set	private_memory	thread_count	properties
⊟ w3wp.exe	10476	123 s	256,836 KB	216,768 KB	71	Properties..
⊟ cmd.exe	55952	<1 s	4,368 KB	3,364 KB	2	Properties..
DaaSRunner.exe	1476	4 s	15,540 KB	11,248 KB	4	Properties..

图 8-9　进程管理

单击右侧的 Properties 按钮，如图 8-10 所示，可以看到进程的详细信息，包括内存、句柄和线程信息。

w3wp.exe:10476 Properties

General　Modules　Handles　Threads

id	10476
name	w3wp
file name	D:\Windows\SysWOW64\inetsrv\w3wp.exe
handle count	2,307
module countid	200
thread count	71
start time	2014-06-03T01:33:47.2516163Z
total cpu time	00:02:03.2031250
user cpu time	00:01:32.1562500
privileged cpu time	00:00:31.0468750
working set	256,836 KB
peak working set	277,996 KB
private memory	216,768 KB
virtual memory	834,212 KB
peak virtual memory	855,968 KB

图 8-10　查看进程信息

8.1.5　REST API

登录到 Kudu 控制台，在首页的下方，如图 8-11 所示，显示 REST API。REST API 基于 JSON。

REST API (works best when using a JSON viewer extension)

- App Settings
- Deployments
- Files
- Processes and mini-dumps
- Runtime versions
- Site Extensions (installed)
- Source control info
- Web hooks
- Web jobs

图 8-11 KUDU REST API

单击 Processes and mini-dump，如图 8-12 所示，列出当前网站的所属进程。

```
[
  - {
        id: 55952,
        name: "cmd",
        href: "https://drumboy.scm.azurewebsites.net/api/diagnostics/processes/55952"
    },
  - {
        id: 1476,
        name: "DaaSRunner",
        href: "https://drumboy.scm.azurewebsites.net/api/diagnostics/processes/1476"
    },
  - {
        id: 10476,
        name: "w3wp",
        href: "https://drumboy.scm.azurewebsites.net/api/diagnostics/processes/10476"
    }
]
```

图 8-12 进程信息 REST API 结果

单击进程的超链接，比如 w3wp.exe 的超链接，如图 8-13 所示，显示该进程的详细信息。这些信息与进程管理功能中的信息是相同的。

```
{
    id: 10476,
    name: "w3wp",
    href: "https://drumboy.scm.azurewebsites.net/api/diagnostics/processes/10476",
    minidump: "https://drumboy.scm.azurewebsites.net/api/diagnostics/processes/10476/dump",
    gcdump: "https://drumboy.scm.azurewebsites.net/api/diagnostics/processes/10476/gcdump",
    parent: "https://drumboy.scm.azurewebsites.net/api/diagnostics/processes/-1",
  - children: [
        "https://drumboy.scm.azurewebsites.net/api/diagnostics/processes/55952"
    ],
  + threads: […],
  + open_file_handles: […],
  + modules: […],
    file_name: "D:\Windows\SysWOW64\inetsrv\w3wp.exe",
    handle_count: 2311,
    module_count: 200,
    thread_count: 71,
    start_time: "2014-06-03T01:33:47.2516163Z",
    total_cpu_time: "00:02:06.7031250",
    user_cpu_time: "00:01:34.5000000",
    privileged_cpu_time: "00:00:32.2031250",
    working_set: 263659520,
    peak_working_set: 285417472,
    private_memory: 223453184,
    virtual_memory: 854429696,
    peak_virtual_memory: 876511232,
    paged_system_memory: 740288,
    non_paged_system_memory: 170254,
    paged_memory: 223453184,
    peak_paged_memory: 245088256
}
```

图 8-13 进程详细信息

单击 minidump 的超链接，可以生成并下载一个包含最基本信息的内存转储文件。如果需要完整的内存转储文件，请使用前面提到的 ProcDump 工具。

8.2　诊断即服务

Azure 网站是一个多租户（multi-tenant）的 PaaS 服务。尽管 Azure 网站提供了极佳的开发体验和诊断功能，但是当部署在 Azure 网站的应用出现问题后，客户始终无法像在本地环境那样收集数据、分析数据和发现问题。另外，Azure 网站客户本身有很多中小企业客户和个人用户。对于这些用户，他们缺乏专业 IT 人员，当网站运行中出现问题时，往往束手无策。

Azure 网站提供了诊断即服务（Diagnostics As A Service，DaaS）功能，该功能提供了两个诊断选项：Diagnose Now 和 Scheduled Analysis。Diagnose Now 立即启动诊断服务。Scheduled Analysis 指定诊断服务在特定的时间运行，对于只在特定时间发生的问题非常有用。

诊断服务启动后，自动收集相关的数据（包括事件日志、IIS 日志、内存转储文件等）。数据收集完成后，利用微软公司技术支持部门的强大的诊断引擎自动分析这些数据，查找可疑问题并提供解决问题的建议。

可以在线查看诊断报告，也可以下载收集的日志和报告到本地计算机进一步研究分析。日志保存在 d:\home\Data\DaaS\Logs 目录下，报告保存在 d:\home\Data\DaaS\Reports 目录下。

在本节中，介绍如何通过诊断即服务功能解决一个真实客户的 WordPress（PHP）网站性能问题。

8.2.1　安装诊断即服务

诊断即服务功能是 Azure 网站的一个扩展模块，客户可以通过 SCM 站点自行安装该模块。

（1）登录到网站的 SCM 网站。

（2）单击顶部菜单栏右边的 Site Extensions。

（3）Site Extension 默认有两个页面：Installed 和 Gallery，选择 Gallery。

（4）如图 8-14 所示，会看到微软公司发布的 Diagnostics as a Service 扩展。单击“+”安装 Diagnostics as a Service。

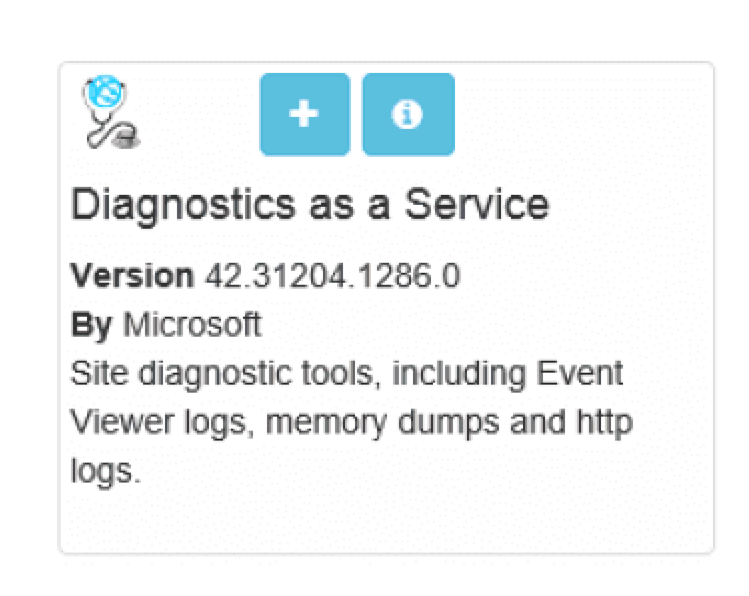


图 8-14　安装诊断即服务扩展

（5）安装完成后，单击页面右上角的 Restart Site，重启 SCM 网站。

8.2.2　使用诊断即服务排查 PHP 性能问题

当 PHP 网站遇到性能问题时，可以通过诊断即服务功能排查问题原因。

（1）登录到网站的 SCM 网站。

（2）单击顶部菜单栏右边的 Site Extensions。

（3）Site Extension 默认有两个页面：Installed 和 Gallery，选择 Installed。

（4）Installed 页面显示所有已经安装的网站扩展。单击 Diagnostics as a Service 扩展模块的“开始”按钮，启动诊断控制台。

（5）诊断控制台提供了两个诊断选项：Diagnose Now 和 Scheduled Analysis。在这里选择 Diagnose Now。

（6）诊断启动后，首先收集相关的信息，然后启动信息分析程序，最后生成分析报告。当完成后，可以看到 Collection Status 和 Analysis Status 都变成 Complete 状态。

（7）单击右面的箭头图标，进入分析报告页面。如图 8-15 所示，分析报告包含如下内容：

- 事件日志。
- W3WP.EXE 内存转储文件。
- HTTP 日志。
- PHP 错误日志。
- PHP 进程分析。

Session: 2014-12-16 07:30:52

Diagnoser	Collection Status	Analysis Status
Event Viewer Logs		
Memory Dump		
Http Logs		
PHP error Logs		
PHP Process Report		

图 8-15　诊断报告

（8）单击 PHP Process Report，打开 PHP 进程分析报告，可以看到如下信息：

① 总结。

Azure 网站中，PHP 是单线程模式。为了支持高并发访问量，IIS 启用多个 PHP-CGI.exe 进程来同时处理多个 HTTP 请求。在总结部分包含当前正在运行的 PHP-CGI.exe 的进程信息和每个 PHP-CGI.exe 正在处理的 HTTP 请求。其中，进程结尾的数字是进程编号。图 8-16 是一个报告总结部分的实例。

Summary

Process	URI
PHP-CGI-2996	/index.php
PHP-CGI-4412	/dbquery.php
PHP-CGI-5616	/index.php
PHP-CGI-7440	/dbquery.php

图 8-16　PHP-CGI 进程总结信息

② 通用信息。

主要包括操作系统版本、进程 CPU 使用率、机器启动时间、进程启动时间等信息。

③ 调用堆栈。

因为 PHP 是解释执行的语言，调用堆栈中不包含 PHP 的调用堆栈。

④ 加载的模块。

⑤ 句柄信息。

⑥ PHP 请求的具体信息，如图 8-17 所示。



图 8-17　PHP 请求具体信息

⑦ 线程 CPU 使用情况。

⑧ PHP 调用堆栈。

调用堆栈在分析问题的时候至关重要。当程序出现异常的时候，调用堆栈可以非常直接地提示报错的函数名称。通过分析调用堆栈及其参数传递和本地变量，可以找到问题的根本原因。当程序出现性能问题时，通常当前正在执行的函数就是导致性能变慢的函数，而调用堆栈可以告诉我们当前正在执行的函数。

PHP 是解释执行语言，与 C 语言等不同，Windows 调试器（WinDbg）本身并不能显示 PHP 调用堆栈。同样，WinDbg 调试器也无法直接显示 C#应用的调用堆栈。为了解决该问题，.NET Framework 提供一个基于 Windows 调试器（WinDbg）的调试扩展模块（SOS.DLL），通过该模块，开发人员可以在 WinDbg 中查看.NET 应用的调用堆栈。在分析 PHP 源代码后，本书作者编写了一个基于 Windows 调试器（WinDbg）的 PHP 调试扩展模块。该模块可以生成当前正在执行的 PHP 页面的调用堆栈。该调试扩展模块集成到了 DaaS 服务中。

如下所示，PHP 的调用堆栈清楚地告诉我们当前正在执行的 PHP 函数及其文件名和当前正在执行的代码行号。根据调用函数名称，可以猜测到 wordPress 的一个插件 captcha 正在进行清理工作。结合下面的 PHP 全局变量信息，很快就能定位到问题的根源。

```
PHP Call Stack:
#   Function Name              File Name
00  cleanup
C:\DWASFiles\Sites\mysite\VirtualDirectory0\site\wwwroot\wp-content\plu
gins\really-simple-captcha\really-simple-captcha.php@251
01  wpcf7_cleanup_captcha_files
C:\DWASFiles\Sites\mysite\VirtualDirectory0\site\wwwroot\wp-content\plu
gins\contact-form-7\modules\captcha.php@418
02  main
C:\DWASFiles\Sites\mysite\VirtualDirectory0\site\wwwroot\wp-content\plu
gins\contact-form-7\modules\captcha.php@439
03  wpcf7 load modules     C:\DWASFiles\Sites\mysite\VirtualDirectory0\
site\wwwroot\wp-content\plugins\contact-form-7\settings.php@36
05  do action              C:\DWASFiles\Sites\ mysite\VirtualDirectory0\
site\ wwwroot\wp-includes\plugin.php@406
06  main                   C:\DWASFiles\Sites\mysite\VirtualDirectory0\
site\wwwroot\wp-settings.php@211
07  main                   C:\DWASFiles\Sites\mysite\VirtualDirectory0\
site\wwwroot\wp-config.php@90
08  main                   C:\DWASFiles\Sites\mysite\VirtualDirectory0\
site\wwwroot\wp-load.php@29
09  main                   C:\DWASFiles\Sites\mysite\VirtualDirectory0\
site\wwwroot\wp-blog-header.php@12
0a  main                   C:\DWASFiles\Sites\mysite\VirtualDirectory0\
site\wwwroot\index.php@17
```

⑨ PHP 全局变量信息。

该区域包含 PHP 的全局变量信息。从中可以看到 WordPress 的 captcha 插件在处理文件的时候遇到了权限问题。

```
PHP Core Globals:
output buffering        : 4096
memory limit            : 134217728
max input time          : 60
track errors            : 0
display_errors          : 0
display_startup_errors  : 0
display startup errors  : 0
log errors              : 1
log errors max len      : 1024
error_log               : D:\home\LogFiles\php_errors.log
```

```
upload tmp dir           : C:\DWASFiles\Sites\mysite\Temp
upload max filesize      : 2097152
file uploads             : 1
variables order          : GPCS
last error type          : E WARNING
last error message:
unlink(C:\DWASFiles\Sites\mysite\VirtualDirectory0\site\wwwroot/wp-cont
ent/uploads/wpcf7_captcha/1696868155.txt): Permission denied
last_error_file
C:\DWASFiles\Sites\mysite\VirtualDirectory0\site\wwwroot\wp-content\plu
gins\really-simple-captcha\really-simple-captcha.php
last_error_lineno       : 251
```

结合 PHP 调用堆栈和 PHP 全局变量，发现 WordPress 的 captcha 插件在处理文件的时候遇到权限错误。检查后发现已经有大量的文件无法删除，而 captcha 每次被调用的时候都会尝试删除所有的临时文件，这导致页面执行时间变长。而由于权限问题无法删除临时文件导致文件数量不断增加，这导致性能进一步变慢。修改相应权限后，问题解决。

8.3　应用配置转换

当在本地 Windows 系统上运行 IIS 时，在 system32\inetsrv\config 目录下有一个 IIS 的配置文件：applicationHost.config。该文件包括所有网站的定义、应用程序池以及全局的默认 Web 服务器的设置（类似于 machine.config 和根 web.config 的.NET 框架的设置）。对于运行在本地的 IIS，所有的网站共享一个 ApplicationHost.config。

Microsoft Azure 网站是一个典型的多租户环境。修改 ApplicationHost.config 中的全局配置会影响运行在相同机器上的所有网站。在 Azure 网站中，为了保证每个站点之间互相独立，Azure 网站为每个站点都生成一个单独的 ApplicationHost.config。Azure 网站优化了很多设置选项，所以默认的 ApplicationHost.config 适用于绝大多数情景。在某些情况下，根据业务需要，可能需要修改默认的配置。但是，这并不是一件容易的事情。

（1）很多全局配置选项被锁定，无法在 web.config 中覆盖全局配置。

（2）Azure 网站并没有提供直接修改 ApplicationHost.config 的功能，无法直接修改该文件。

Azure 网站通过 ASP.NET 4 中的 web.config XML-Document-Transformation（XDT）技术解决了该问题。关于 XDT 技术，可以参考以下文档：

http://msdn.microsoft.com/zh-cn/library/dd465326(v=vs.100).aspx

XDT 引擎是一个开源项目，可以在 Nuget 网站下载安装：

https://www.nuget.org/packages/Microsoft.Web.Xdt

8.3.1　XDT 简介

使用 XDT 转换技术，需要创建一个转换文件。转换文件是一个 XML 文件，该文件指

定如何更改 ApplicationHost.config。转换操作通过使用在 XML-Document-Transform 命名空间（映射到 xdt 前缀）中定义的 XML 特性来指定。

XML-Document-Transform 命名空间定义了两个属性：Locator 和 Transform。Locator 属性指定要以某种方式更改的 ApplicationHost.config 元素或一组元素。Transform 属性指定要对 Locator 属性所查找的元素执行哪些操作。

Locator 属性告诉转型引擎如何找到元素或设置要更改/转换元素。支持的方式如下：

- 精确匹配某个节点的属性的值。
- 精确的 XPath。
- 条件匹配。

Transform 属性告诉转换引擎对于查找到的元素进行哪些操作。支持的操作如下：

- 替换节点。
- 插入一个节点。
- 删除一个节点。
- 删除属性。
- 设置属性。

8.3.2 通过 XDT 转换修改 ApplicationHost.config

使用 XDT 转换修改 ApplicationHost.config，需要遵循以下步骤：

（1）创建一个转换文件，命名为 ApplicationHost.xdt。

（2）上传至 Azure 网站的/site 目录下。

```
/site
    applicationhost.xdt
    /wwwroot
```

（3）重新启动网站。

8.3.2.1 配置转换环境变量

在配置转换中，可以使用下面的环境变量来定位某些元素。

- XDT_SITENAME：网站名称。
- XDT_SCMSITENAME：SCM 网站名称。
- XDT_EXTENSIONPATH：应用程序扩展的路径。
- HOME：网站的根目录，通常是 D:\HOME。

8.3.2.2 配置转换实例 1：修改 FastCGI maxInstances 配置

在 Azure 网站中，默认的 FastCGI 的最大实例数（maxInstances）设置为 15。对于运行在大型实例的网站（4 核/8GB 内存），这个设置无法充分利用系统硬件资源。通常，推荐设置 maxInstances 为 10*#CPU。下面的转换配置文件（ApplicationHost.xdt）将 PHP 的最大实例数增加到 40：

```
<?xml version="1.0"?>
<configuration xmlns:xdt="http://schemas.microsoft.com/XML-Document-Transform">
  <system.webServer>
    <fastCgi>
      <application xdt:Locator="Match(fullPath)" xdt:Transform="SetAttributes
      (maxInstances)"
           fullPath="D:\Program Files (x86)\PHP\v5.4\php-cgi.exe"
           maxInstances="40"/>
    </fastCgi>
  </system.webServer>
</configuration>
```

8.3.2.3　配置转换实例 2：添加环境变量

下面的转换配置文件（ApplicationHost.xdt）添加一个环境变量。该环境变量指定 Node 采用 Visual Studio 2012 版本的 MSBUILD.EXE 来编译使用 Visual C++ 2005 开发的 Node 模块。

```
<?xml version="1.0"?>
<configuration xmlns:xdt="http://schemas.microsoft.com/XML-Document-Transform">
    <system.webServer>
        <runtime xdt:Transform="Insert">
           <environmentVariables>
              <add name=" GYP MSVS VERSION " value="2012"/>
           </environmentVariables>
        </runtime>
    </system.webServer>
</configuration>
```

8.3.2.4　使用 CTT 工具在本地测试转换

转换配置并不是一件容易的工作，通常需要尝试几次才能够获得希望的结果。CTT 是一个基于 XDT 转换引擎的开源工具。使用 CTT 工具，可以在本地测试转换结果，极大地简化了配置转换工作。

可以在 CodePlex 网站下载 CTT 工具：

http://ctt.codeplex.com/

下面的步骤演示了如何使用 CTT 工具在本地进行配置转换测试工作。

（1）使用 Kudu 控制台下载网站的 ApplicationHost.config 文件。关于如何使用 Kudu 控制台下载文件，请参考 8.1.3 节。

（2）创建配置转换文件 ApplicationHost.xdt 文件。

（3）运行 CTT 工具测试转换。其中，s 指定源配置文件，t 指定配置转换文件，d 指定转换结果，v 指定输出详细信息。

```
ctt s:"applicationHost.config" t:"applicationhost.xdt" d:"result.xml" v
```

（4）图 8-18 是一个 CTT 运行的实例。

```
Executing SetAttributes (transform line 6, 110)
on /configuration/system.applicationHost/sites/site[@name='xdttest']/traceFailed
RequestsLogging
Applying to 'traceFailedRequestsLogging' element (no source line info)
File: , LineNumber: 6, LinePosition: 110, Message: Argument 'enable' did not mat
ch any attributes
File: , LineNumber: 6, LinePosition: 110, Message: No attributes found to set
Done executing SetAttributes
Executing Remove (transform line 15, 86)
on /configuration/system.webServer/tracing/traceFailedRequests/add/failureDefini
tions[@statusCodes='400-600']
Applying to 'failureDefinitions' element (no source line info)
Removed 'failureDefinitions' element
Done executing Remove
Executing Insert (transform line 16, 74)
on /configuration/system.webServer/tracing/traceFailedRequests/add/failureDefini
tions
Applying to 'add' element (no source line info)
Inserted 'failureDefinitions' element
Done executing Insert
```

图 8-18　CTT 命令行

（5）如果 CTT 测试通过，可以打开结果文件检查配置转换结果是否与期望的一致。

8.3.2.5　Azure 网站配置转换日志

Azure 网站在转换配置文件时生成一个日志文件，可以帮助用户定位配置转换相关的问题。该文件位于/site/LogFiles/Transform 目录下，如图 8-19 所示。

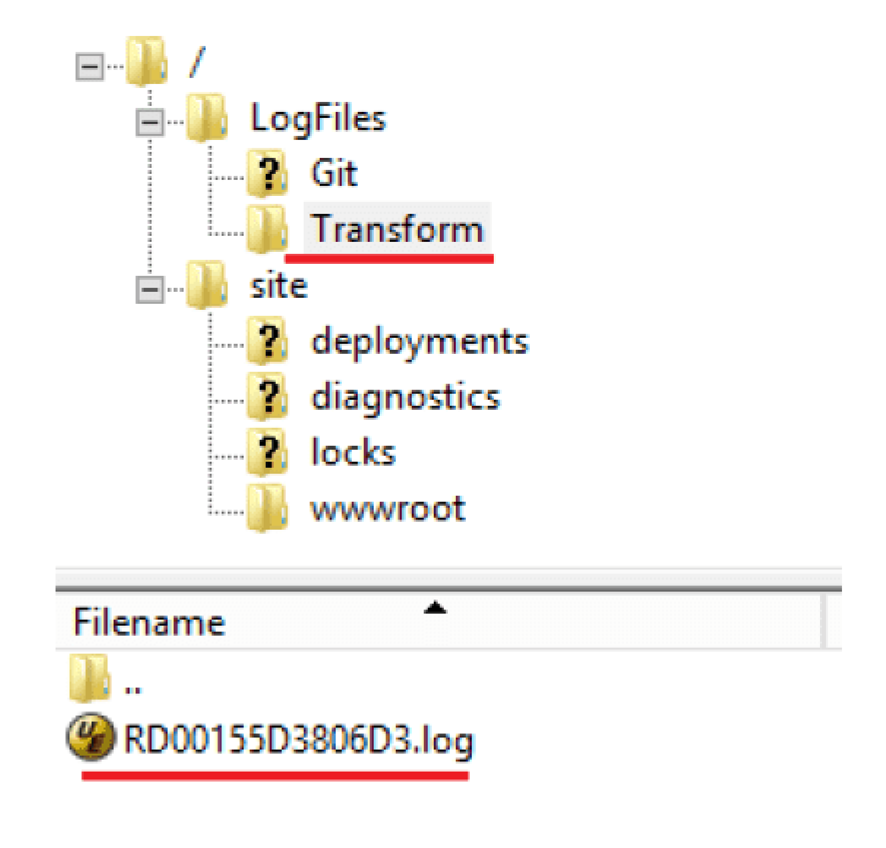


图 8-19　配置转换日志

下面是一个配置转换日志的实例：

```
    2014-05-27T06:14:38 :(15,12), Microsoft.Web.XmlTransform.XmlNodeException:
No attribute 'name' exists for the Match Locator ---> Microsoft.Web.XmlTransform.
XmlTransformationException: No attribute 'name' exists for the Match Locator
      at Microsoft.Web.XmlTransform.Match.ConstructPredicate()
      at Microsoft.Web.XmlTransform.Locator.ConstructPath()
      at Microsoft.Web.XmlTransform.Locator.ConstructPath(String parentPath,
XmlElementContext context, String argumentString)
```

```
    at Microsoft.Web.XmlTransform.XmlElementContext.ConstructXPath()
```

8.3.2.6　注意事项

（1）配置转换发生在网站启动之前，因此在修改 ApplicationHost.xdt 后，需要重新启动网站才能使配置转换生效。

（2）假如在配置转换时发生错误，可能无法访问网站。

8.4　最 佳 实 践

Microsoft Azure 网站使用户能够在 Microsoft Azure 上快速构建高度可扩展的网站。本节讨论一些最佳实践。这些最佳实践有助于读者更好地利用 Microsoft Azure 网站提供的基础设施，并为最终用户提供一个高性能、可靠的网站。

8.4.1　设计一个可扩展的架构

架构中的瓶颈可以导致严重的性能问题。比如，要访问的某个资源性能较差，此时客户请求就会被阻塞。而且，此类性能瓶颈无法通过增加多个实例解决。

在架构中要尽量避免引入单点故障。比如，如果应用依赖于数据库，那么当数据库出现故障时，整个应用就会瘫痪。数据库就是架构中的单点故障。

要避免上面的数据库单点故障，在架构设计时，需要考虑以下两点：

（1）数据库的冗余与同步。必须考虑建立一个备用数据库，并且实时将生成环境数据库同步到备用数据库。

（2）应用设计。当出现生产环境数据库故障时，应用能够检测到故障，并自动切换到备用数据库。

8.4.2　设计一个灵活应变的架构

提高应变能力和快速响应是业务成功的重要因素。然而，设计满足业务弹性需求的 IT 架构设计是具有挑战性的任务。一个好的、有弹性的架构能够保证业务的连续性和可用性，并且考虑灾难恢复策略。

8.4.3　合理利用其他 Azure 服务

Azure 提供了多种服务，可以帮助开发者提高应用性能，实现自动故障转移等。

比如，流量管理器可以帮助开发者实现以下目标：

- 提高关键应用的可用性。
- 提高高性能应用程序的响应能力。

- 零停机网站维护和升级。

Microsoft Azure 内容传送网络（CDN）通过在遍布美国、欧洲、亚洲、澳大利亚和南美洲的众多物理节点上缓存静态内容，提供一个传送高带宽内容的全球性解决方案。通过 CDN 可以降低网络延迟，提高响应速度。使用 CDN，您的网站在处理瞬时高负荷时会提供更快的响应。

Microsoft Azure 缓存服务（预览版）是全新的缓存解决方案。通过 Microsoft Azure 缓存服务，可以在 Microsoft Azure 中构建快速且可缩放的应用程序。Microsoft Azure 缓存服务有助于应用程序即使在用户负载增加的情况下也能迅速地做出响应。单独的分布式缓存层可以更加高效地使用应用层的计算资源。

8.4.4 合理利用地理冗余

可以将网站部署在遍布全球的数据中心，即使某个数据中心出现故障，网站仍然可以正常运作。

图 8-20 描述了一个典型的高性能、可扩展、高容错的 Azure 网站部署方案。

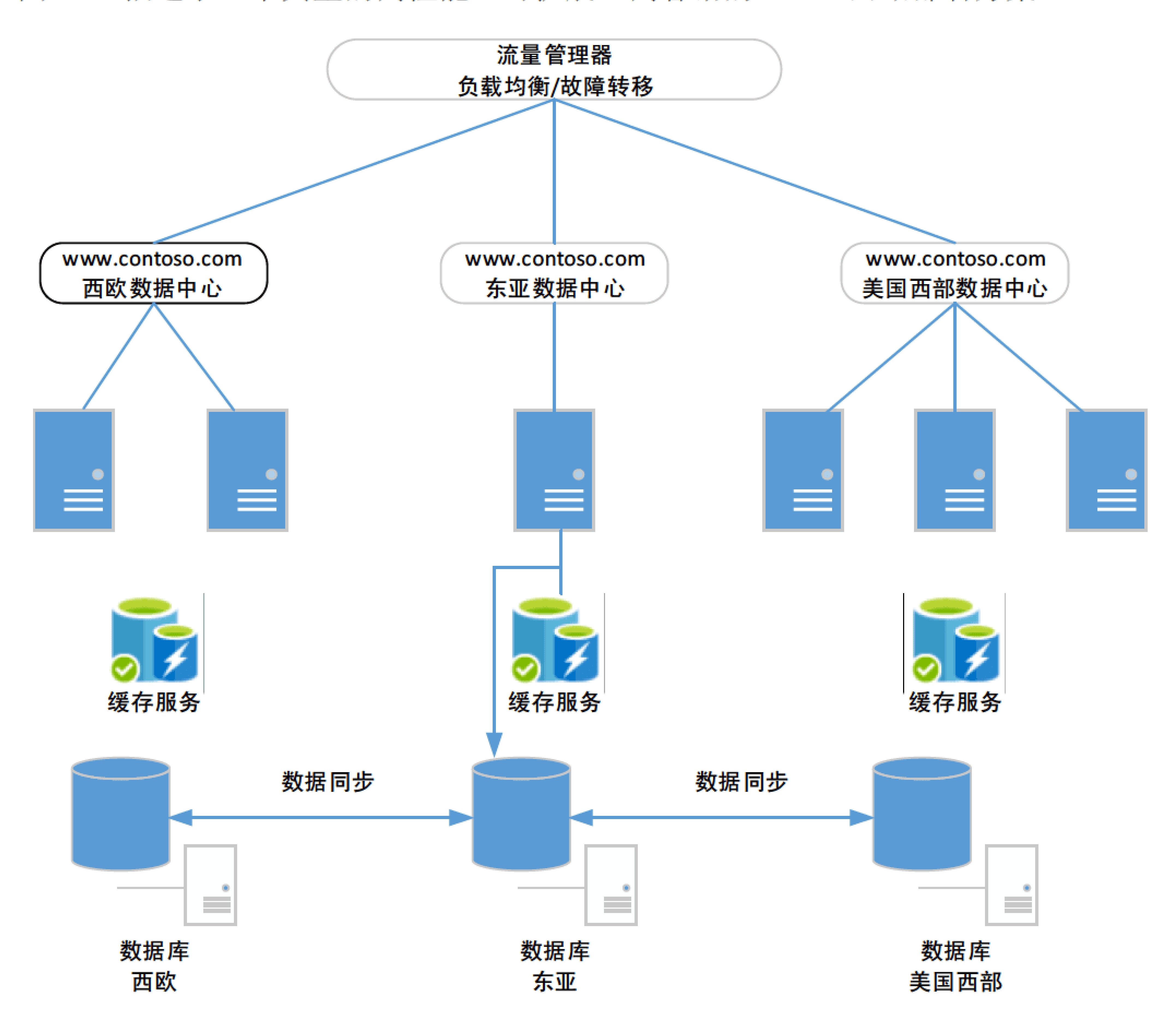


图 8-20 高性能、可扩展、高容错的 Azure 网站部署架构

8.4.5 选择合适的缩放方案

根据业务需要，可以选择根据计划日程缩放网站。比如，如果网站在周末业务繁忙，那么可以设置在周末使用 2 个机器实例，工作日使用 1 个机器实例。

还可以根据 CPU 使用率自动缩放网站。当网站 CPU 使用率超过指定的阈值后，Azure 网站会自动增加一个机器实例；当 CPU 使用率降低后，自动释放空闲机器实例。

8.4.6 及时备份网站

根据业务需要，选择合适的网站备份方案。当出现灾难性的问题时可以及时恢复网站。

8.4.7 配置动态 IP 限制功能

通常，运行在本地的 IIS 服务器都具有防止拒绝服务攻击的防火墙。在 Azure 网站上不能控制防火墙，但是可以通过 IP 限制功能拒绝某些特定 IP 地址的攻击请求。也可以配置动态 IP 限制功能，自动拒绝疑似攻击。IP 限制模块和动态 IP 限制模块都是 IIS 本身自有的功能。建议阅读下面的文档来更好地了解这两个模块：

Using Dynamic IP Restrictions

http://www.iis.net/learn/manage/configuring-security/using-dynamic-ip-restrictions

IP Address and Domain Restrictions

http://technet.microsoft.com/en-us/library/hh831785.aspx

动态 IP 地址模块具有下面的功能：

（1）根据并发请求数阻止 IP 地址。如果一个 HTTP 客户端超过允许的并发请求数，该客户端的 IP 地址被暂时封锁。

（2）基于一段时间的请求数封锁 IP 地址。如果一个 HTTP 客户端在指定时间内发送了超过允许的请求，该客户端的 IP 地址被暂时封锁。

（3）白名单功能。

（4）丰富的拒绝操作选择。该模块可以返回 HTTP 403、HTTP 404 或者只是终止 HTTP 连接，不返回任何响应。

（5）支持使用代理服务器，支持 IPv6。

下面是一个动态 IP 限制模块的场景配置。它定义了拒绝服务的动作和两个拒绝访问的条件。当条件满足时，拒绝服务的动作被执行。下面的例子中定义了两个条件，当任意一个条件满足时，定义的动作都会被执行（中断请求动作 AbortRequest，直接断开 TCP 连接）。

（1）单个用户同时超过 10 个请求在执行，则第 11 个请求会被拒绝。

（2）当某个用户在 20 秒内发送了超过 50 个请求，则该用户的后续请求会被拒绝，直到请求强度低于每 20 秒 50 个请求。

```
<?xml version="1.0" encoding="utf-8" ?>
<configuration>
  <system.webServer>
    <security>
      <dynamicIpSecurity denyAction="AbortRequest" enableProxyMode="false">
        <denyByConcurrentRequests enabled="true" maxConcurrentRequests="10"/>
        <denyByRequestRate enabled="true"
                 maxRequests="50" requestIntervalInMilliseconds="20000" />
      </dynamicIpSecurity>
    </security>
  </system.webServer>
</configuration>
```

8.4.8 配置自我修复功能

很多开发工程师或者技术支持工程师都有深夜被叫醒处理紧急应用问题的经历。根据统计，很多应用问题可以通过简单地重启网站来解决。Microsoft Azure 网站（WAWS）试图通过自动修复功能解决这些问题。该功能只在标准模式下可用。

自动修复功能自动检测定义的错误条件。当条件满足时，Azure 网站自动执行定义的修复措施，比如自动重启网站。

自动修复功能规则由两部分组成：触发器（trigger）和修复措施（action）。

1. 触发器

自动修复功能支持 4 种触发器：

（1）HTTP 请求数：在一定时间段内指定的请求数超过指定的数目。可以用于指定并发请求数，比如，一秒内超过 500 个请求。当然，也可以指定更长时间段，比如 1 小时内的请求数量。当超过该数量时，执行指定的修复措施。

（2）HTTP 状态码：HTTP 响应的代码。比如 1 分钟内有超过 100 个请求的响应为 500（服务器端错误），则执行指定的修复措施。

（3）内存使用量：IIS 工作进程的内存使用情况。比如，IIS 工作进程的内存使用量超过 1GB 则执行指定的修复措施。

（4）缓慢的 HTTP 请求数：在一定时间段内，响应时间超过指定时间的请求数目。比如，在 10 分钟内，有 500 个请求响应时间超过 20 秒，则执行指定的修复措施。

2. 修复措施

自我修复功能支持 3 种修复措施：

（1）重启 IIS 工作进程：重新启动当前的 IIS 工作进程，当重启时，在事件日志中会记录 ID 为 2299 的事件。

（2）记录到事件日志：只记录一条 ID 为 9020 的信息到事件日志。

（3）自定义：执行客户指定的程序。

3. 自动修复功能的定义

自动修复功能的定义如下：

```
<configSchema>
    <sectionSchema name="system.webServer/monitoring">
      <element name="triggers">
        <element name ="requests">
            <attribute name="count" type="uint" defaultValue="0" />
            <attribute name="timeInterval" type="timeSpan" defaultValue=
            "00:00:00" />
        </element>
        <element name ="statusCode">
          <collection addElement="add">
            <attribute name="statusCode" type="uint" defaultValue="0" />
            <attribute name="subStatusCode" type="uint" defaultValue="0" />
            <attribute name="win32StatusCode" type="uint" defaultValue="0" />
            <attribute name="count" type="uint" defaultValue="0" />
            <attribute name="timeInterval" type="timeSpan" defaultValue=
            "00:00:00" />
          </collection>
        </element>
        <element name="memory">
            <attribute name="privateBytesInKB" type="uint" defaultValue="0" />
        </element>
        <element name="slowRequests">
          <attribute name="timeTaken" type="timeSpan" defaultValue="00:00:30"/>
          <attribute name="count" type="uint" defaultValue="0" />
          <attribute name="timeInterval" type="timeSpan" defaultValue=
          "00:00:00" />
        </element>
      </element>
      <element name ="actions">
        <attribute name="value" type="enum" defaultValue="Recycle">
          <enum name="Recycle" value="0" />
          <enum name="LogEvent" value="1" />
          <enum name="CustomAction" value="2" />
        </attribute>
        <element name="customAction" >
          <attribute name="exe" type="string" />
          <attribute name="parameters" type="string" />
        </element>
      </element>
    </sectionSchema>
</configSchema>
```

下面介绍几个比较常见的场景。

1）一定时间内请求数量超过设置则记录一条日志信息

考虑这样一个场景，应用可能有扩展性问题。在处理大量并发请求时，可能会出现问题。但是还需要一段时间来找到泄漏原因，并修复问题。在此之前，当规定时间内客户请求数超过一定数量时，需要重启网站以避免问题。

下面的 web.config 可以实现该需求，如果网站已经有 web.config，那么需要将 monitoring 部分合并到已有的 web.config 中。

```
<?xml version="1.0" encoding="utf-8" ?>
<configuration>
  <system.webServer>
    <monitoring>
      <triggers>
        <requests count="1000" timeInterval="00:00:05" />
      </triggers>
      <actions value="LogEvent" />
    </monitoring>
  </system.webServer>
</configuration>
```

上面的配置表示：如果 5 秒钟内的请求数超过 1000 个，在事件日志中记录一条信息。该配置以 5 秒钟为计时间隔，如果在计时器超时以前已经达到 1000 个请求，则在事件日志中记录一条信息；如果在计时器超时时没有达到 1000 个请求，计时器和请求数都将被归零，重新开始计数。当触发器被触发时，会在日志中看到编号为 9020 的日志信息。

```
<Event>
  <System>
    <Provider Name="IIS-Process-Monitoring"/>
    <EventID>9020</EventID>
    <Level>0</Level>
    <Task>0</Task>
    <Keywords>Keywords</Keywords>
    <TimeCreated SystemTime="12:18:57 PM"/>
    <EventRecordID>489445359</EventRecordID>
    <Channel>Application</Channel>
    <Computer>RD00155D380B59</Computer>
    <Security/>
  </System>
  <EventData>
    <Data>Worker Process serving application pool 'drumboy' hit the 'Total
    Requests' limit.</Data>
  </EventData>
</Event>
```

2）根据性能情况重启网站

某些情况下，网站可能会响应缓慢，性能变得很差。在问题修复以前，可能希望自动

重启网站以恢复正常状态。下面的 web.config 可以实现该需求，如果网站已经有 web.config，那么需要将 monitoring 部分合并到已有的 web.config 中。

```
<?xml version="1.0" encoding="utf-8" ?>
<configuration>
  <system.webServer>
    <monitoring>
      <triggers>
        <slowRequests timetaken="00:00:15" count="20" timeInterval=
        "00:02:00" />
      </triggers>
      <actions value="Recycle" />
    </monitoring>
  </system.webServer>
</configuration>
```

在该配置中，当监测到在过去 2 分钟内有超过 20 个请求执行了 15 秒以上时，自动重启 IIS 工作进程。请注意，timeInterval 需要大于 timeTaken。当触发器被触发时，除了重启 IIS 工作进程外，还会在事件日志中记录下面的信息：

```
<Event>
  <System>
    <Provider Name="IIS-Process-Monitoring"/>
    <EventID>9020</EventID>
    <Level>0</Level>
    <Task>0</Task>
    <Keywords>Keywords</Keywords>
    <TimeCreated SystemTime="12:32:43 PM"/>
    <EventRecordID>663084312</EventRecordID>
    <Channel>Application</Channel>
    <Computer>RD00155D380B59</Computer>
    <Security/>
  </System>
  <EventData>
    <Data>Worker Process serving application pool 'drumboy' hit the 'Slow
    Requests' limit.</Data>
  </EventData>
</Event>
```

3）根据 HTTP 错误代码重启网站

有些时候，网站可能进入了错误状态。此时，网站开始返回 HTTP 500 错误。同样，在问题修复以前，下面配置可以在问题发生时自动重启 IIS 工作进程。如果网站已经有 web.config，那么需要将 monitoring 部分合并到已有的 web.config 中。

```
<?xml version="1.0" encoding="utf-8" ?>
<configuration>
```

```
  <system.webServer>
    <monitoring>
      <triggers>
        <statusCode>
          <add statusCode="500" count="10" timeInterval="00:01:00" />
        </statusCode>
      </triggers>
      <actions value="Recycle" />
    </monitoring>
  </system.webServer>
</configuration>
```

上面的配置定义了基于 HTTP 状态码的触发器。如果在 1 分钟内有 10 个请求发生 500 错误，则重启 IIS 工作进程。同时，在事件日志中记录下面的信息：

```
<Event>
  <System>
    <Provider Name="W3SVC-WP"/>
    EventID>2299</EventID>
    <Level>3</Level>
    <Task>0</Task>
    <Keywords>Keywords
    </Keywords>
    <TimeCreated SystemTime="1:35:07 PM"/>
    <EventRecordID>494015171</EventRecordID>
    <Channel>Application</Channel>
    <Computer>RD00155D380B59</Computer>
    <Security/>
  </System>
  <EventData>
    <Data>Worker Process requested recycle due to 'Status Code' limit.</Data>
  </EventData>
</Event>
```

4）根据内存使用情况重启网站

如果网站有内存泄漏问题，自动修复功能可以在内存泄漏到一定程度的时候自动重启 IIS 工作进程。这样，网站在内存不足之前就被重启，可以避免客户访问网站时遇到问题。

下面配置可以在工作进程内存使用达到 800MB 时自动重启 IIS 工作进程。请注意，800MB 是指某个工作进程的 privateBytes，而不是整个系统的内存使用情况。privateBytes 是指当前进程申请的而且不能与其他进程共享的内存，它是判断是否发生内存泄漏的最主要的指标。如果网站已经有 web.config，那么需要将 monitoring 部分合并到已有的 web.config 中。

```
<?xml version="1.0" encoding="utf-8" ?>
```

```
<configuration>
  <system.webServer>
    <monitoring>
      <triggers>
        <memory privateBytesInKB="819200" />
      </triggers>
      <actions value="Recycle" />
    </monitoring>
  </system.webServer>
</configuration>
```

当定义的内存使用触发器被触发时，IIS 工作进程被重启，而且会在事件日志中记录下面的信息：

```
<Event>
  <System>
    <Provider Name="W3SVC-WP"/>
    <EventID>2299</EventID>
    <Level>3</Level>
    <Task>0</Task>
    <Keywords>Keywords</Keywords>
    <TimeCreated SystemTime="2:08:30 PM"/>
    <EventRecordID>496018921</EventRecordID>
    <Channel>Application</Channel>
    <Computer>RD00155D380B59</Computer>
    <Security/>
  </System>
  <EventData>
    <Data>Worker Process  requested recycle due to 'Memory' limit.</Data>
  <EventData>
</Event>
```

8.4.9　采用多租户模式节约系统资源

很多网站应用，比如 Orchard、WordPress 等，提供了多租户（多站点）模式。启用多站点模式允许使用同一个网站应用运行多个网站。如果网站的业务对价格比较敏感，希望降低成本，则应采用多租户模式。

图 8-21 描述了 Orchard 网站的多站点模式。在这种模式下，一个 IIS 工作进程加载一份 Orchard 运行时库文件、一份 ASP.NET 运行时库文件和 IIS 的模块就可以运行两个或者更多个 Orchard 网站。

从资源使用率的角度来看，多站点模式可以在一个单独的应用程序域中运行多个独立的网站。使用相同的硬件资源，可以运行更多的站点，从而降低成本。

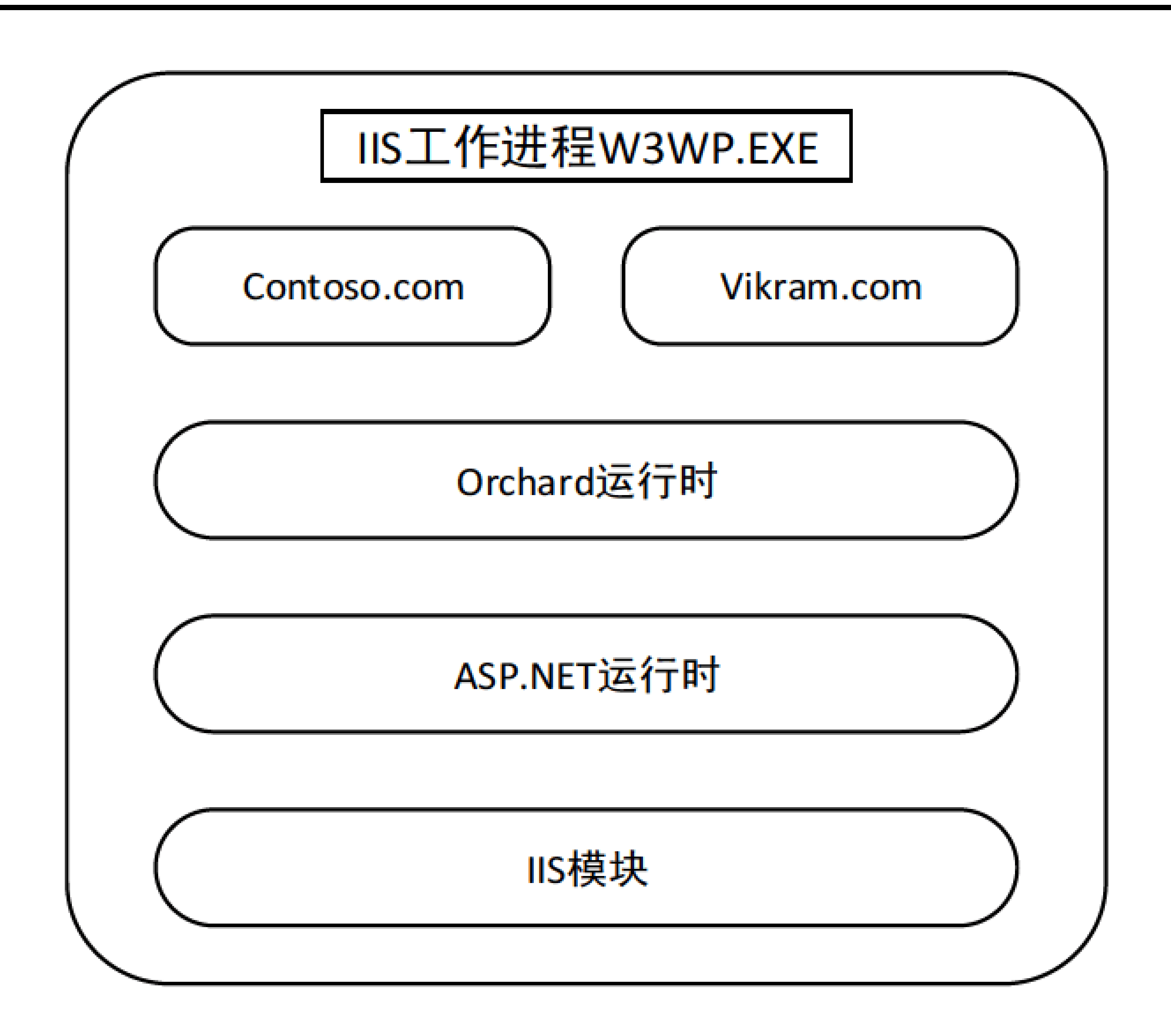


图 8-21 Orchard 多站点模式

图 8-22 描述了单站点模式的架构。显而易见，多站点模式下，多个站点共享很多系统资源，因此单站点模式比多站点模式需要更多的系统资源。当然，如果更关注网站的性能，而不是资源消耗和成本，那么需要选择单站点模式。



图 8-22 单站点模式

8.5 参考文献与扩展阅读

1. Azure 网站最佳实践

基于 Azure 网站，可以运行高扩展性、高可靠性的网站。该文章介绍了 Azure 网站架

构的最佳实践，这些最佳实践可以帮助开发者构建高性能、可扩展的网站。

http://azure.microsoft.com/blog/2014/02/10/best-practices-windows-azure-websites-waws/

2. 在 Azure 网站上运行 WordPress

利用 Azure 网站的基础设施，可以运行可扩展、安全的企业级 WordPress 网站。该文章介绍了如何基于 Azure 网站创建、配置 WordPress 网站和提高 WordPress 网站性能，并且介绍了 Azure 网站上运行 WordPress 的最佳实践。

http://azure.microsoft.com/blog/2014/05/13/how-to-run-wordpress-site-on-azure-websites/

3. 配置多租户 Orchard 网站

Orchard 网站支持多租户模式。基于多租户模式，可以在一个应用程序域中运行多个 Orchard 网站。在 Azure 网站中，这可以节省系统资源，提高系统利用率。

http://docs.orchardproject.net/Documentation/Setting-up-a-multi-tenant-orchard-site

4. Transform your Microsoft Azure Web Site

Azure 网站是一个多租户托管环境。Azure 网站为每个站点自动生成一个 applicationhost.config 配置文件。该 applicationhost.config 包含已经优化的默认设置，适用于大多数的网站。在某些特定的情况下，可能要更改应用要求的默认设置。但是，无法直接编辑配置文件。通过强大的 XML 文档转换功能，就可以修改任何网站配置。

http://www.waws.cn/122

5. Auto-Healing Windows Azure Web Sites

很多情况下，当 Web 应用遇到问题时，希望快速恢复网站服务。利用 Azure 网站的自动修复功能，可以指定网站在遇到特定问题时通过重启等操作尝试自我修复。

http://azure.microsoft.com/blog/2014/02/06/auto-healing-windows-azure-web-sites/

6. Kudu 网站

Kudu 是 Azure 网站 Git 部署的引擎。同时 Kudu 网站提供了非常实用、强大的功能。通过 Kudu 网站，可以浏览网站的文件、运行命令、使用网站诊断 REST API 等。

https://github.com/projectkudu